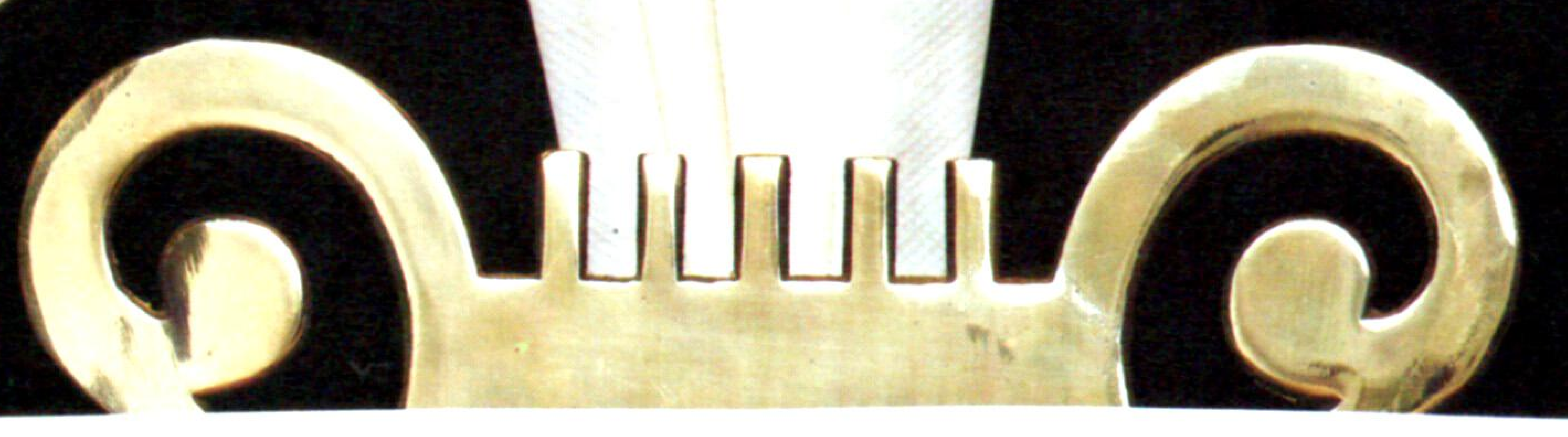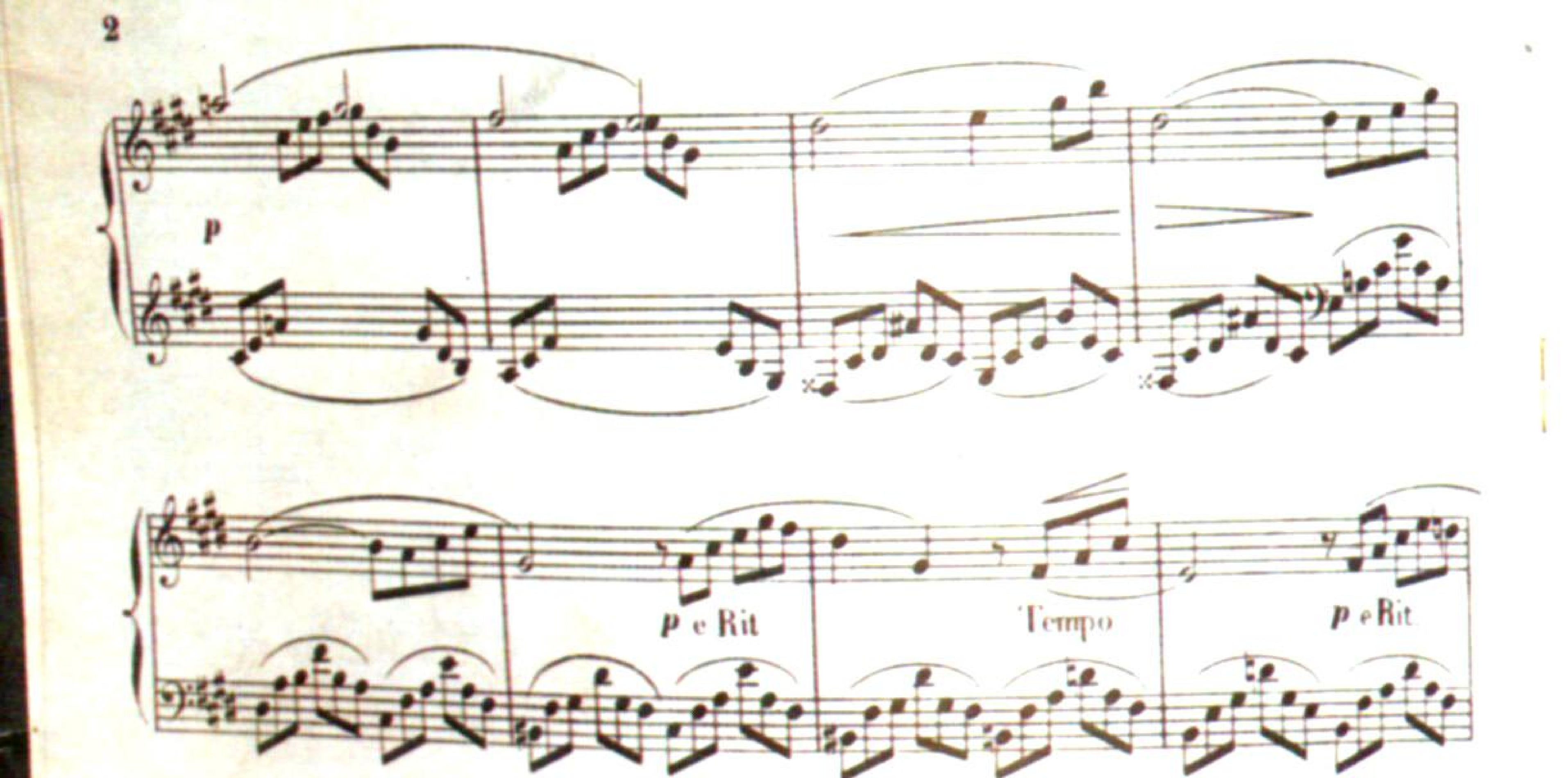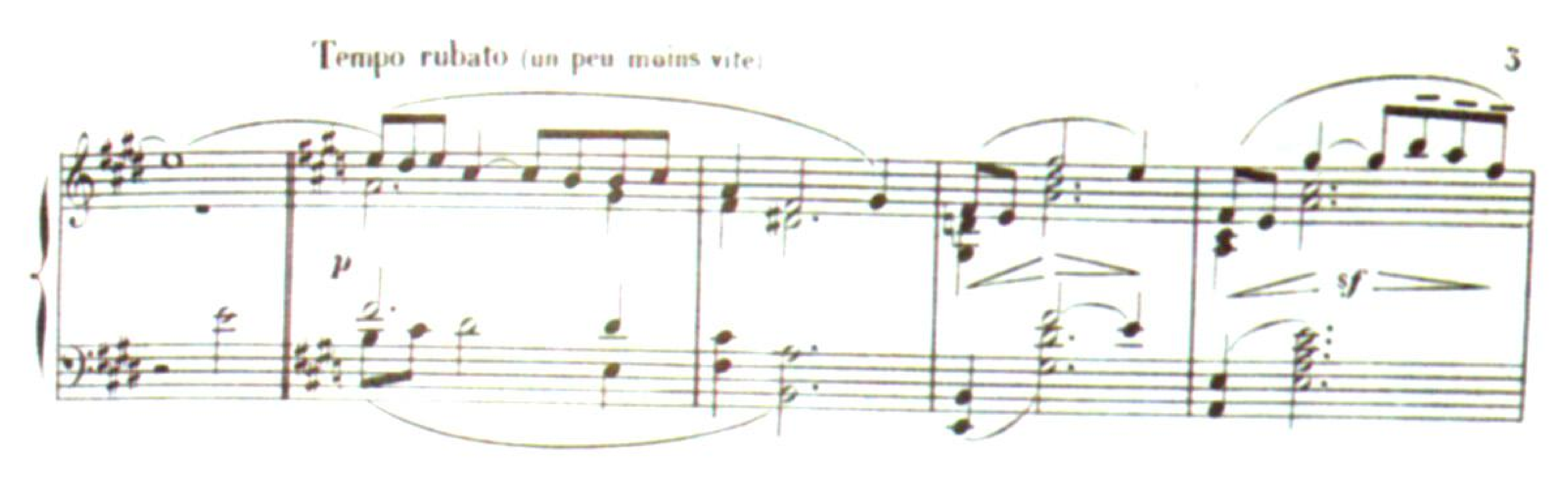
mi computer

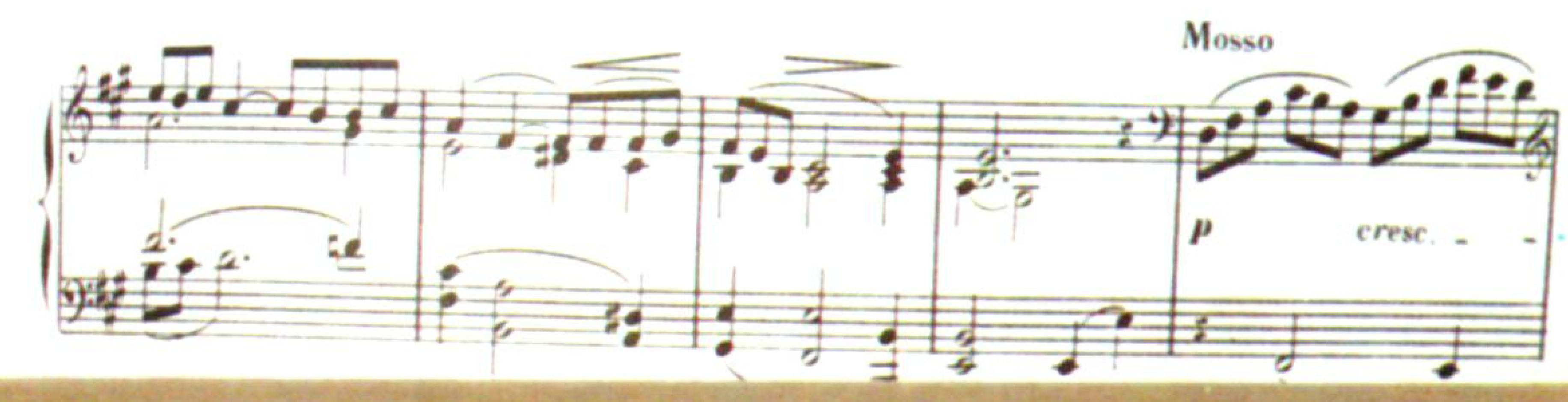
49

CURSO PRAC[...]DOR PERSONAL,
EL MI[...]ENADOR











150 ptas.

Editorial Delta, S.A.

MI COMPUTER CURSO PRACTICO DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen V - Fascículo 49

Director: José Mas Godayol
Director editorial: Gerardo Romero
Jefe de redacción: Pablo Parra
Coordinación editorial: Jaime Mardones
Francisco Martín
Asesor técnico: Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.°, 08008 Barcelona
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-7598-007-4 (tomo 5)
84-85822-82-X (obra completa)
Depósito Legal: B. 52/1984

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 198412
Impreso en España - Printed in Spain - Diciembre 1984

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Argentina: Viscontea Distribuidora, S.C.A., La Rioja 1134/56, Buenos Aires.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 16 690 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 087 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S.A. (Paseo de Gracia, 88, 5.°, 08008 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

No se efectúan envíos contra reembolso.





Creación de melodías

Éste es el primer capítulo de una serie en la que analizaremos el MIDI (interface digital para instrumentos musicales)

La música y las ciencias exactas guardan una relación que se conoce desde hace muchos siglos. El matemático griego Pitágoras pesó un grupo de martillos de herrero para averiguar por qué parecían sonar de forma "melodiosa" al golpear contra el yunque. Descubrió que un martillo de la mitad de peso que otro producía un sonido de exactamente el doble de frecuencia, o una octava más alto. Con ello se estableció el primer principio que rige las relaciones de frecuencia en música.

En la Edad Media los compositores llenaban las catedrales con el sonido de misas y motetes (composiciones corales polifónicas) que estaban proporcionados rítmica y numéricamente con el mismo nivel de precisión que la arquitectura de las propias catedrales. A menudo su música era tan compleja que se creía que sólo los oídos de Dios eran capaces de apreciar las relaciones numéricas, mientras los meros humanos escuchaban música. Y cualquiera que haya presenciado una función de música en vivo (de casi cualquier tipo) habrá observado que los músicos cuentan "1,2,3,4; 2,2,3,4", en un susurro, antes de comenzar a tocar.

Por tanto, fue muy natural que los mundos de la informática y la música se superpusieran. En la actualidad, un desarrollo que está causando gran expectación es el MIDI (*Musical Instrument Digital Interface*: interface digital para instrumentos musicales). Esta unidad está diseñada para permitir que cualquier sistema digital, incluyendo microordenadores, controle las funciones de otro. Como la mayoría de los instrumentos musicales electrónicos que se están fabricando en la actualidad son digitales, se abre a los usuarios de un micro personal todo un nuevo campo de posibilidades.

Pero el MIDI no es una caja mágica. No convierte de la noche a la mañana al usuario de un micro en un Vangelis o un Stevie Wonder. Los conocimientos musicales y la imaginación siempre producen los mejores resultados, tanto si la música se ejecuta en un banco de sintetizadores conectados en interface como en una guitarra acústica.

Para comprender la clase de instrumentos musicales para que el MIDI está pensado y cómo se produce la música electrónica, hemos de remontarnos medio siglo atrás. Ya antes de la segunda guerra mundial los músicos habían empezando a experimentar con *generadores sine-tono* simples. Éstos eran dispositivos eléctricos que hacían vibrar una tira metálica, produciendo de ese modo un tono constante que podía variar en altura. Este sonido se solía utilizar para la ambientación musical de las películas de ciencia-ficción en la década de los cincuenta, para sugerir una atmósfera misteriosa o futurista. Aún hoy se escucha en los altavoces de los televisores cómo una señal para que los telespectadores apaguen los aparatos cuando acaba la emisión. Los primeros órganos Hammond que se comercializaron en los años treinta eran electrónicos y utilizaban esta clase de sonido.

Pero fue el *boom* de la electrónica que se produjo durante la segunda guerra mundial, específicamente el desarrollo de la grabadora de cinta por parte de los alemanes, lo que posibilitó que los músicos crearan y manipularan el sonido de forma bastante diferente. Esto se podía realizar empalmando cinta magnetofónica analógica en la cual ya se había grabado sonido, ya fuera "musical" o de cualquier otro tipo. Estos diminutos retazos de cinta se combinaban entonces esmeradamente para producir un collage de eventos sonoros. Esta "nueva música" representó una ruptura con todas las reglas escritas acerca de teoría musical convencional. Fascinaba a algunos oyentes en la misma medida en que desagradaba a otros.



Marcus Wilson-Smith

Música rupturista
La nueva música exige una nueva notación. Las partituras de Stockhausen, con sus representaciones pictóricas de los sonidos y las directrices gráficas de sincronización, no tienen ninguna relación con las partituras clásicas y, en realidad, fueron pensadas para que se parecieran a los diagramas de circuitos eléctricos

"Doctor" pionero



El tema más clásico de la música electrónica es, sin ninguna duda, *"Doctor Who"*, escrito para el BBC Radiophonic Workshop en 1962 por Ron Grainer (al que vemos a la izquierda de la fotografía, bromeando con algunos de sus compañeros de plató en la serie *Maigret* de la BBC Television)

Al mismo tiempo se fueron difundiendo y haciendo más controlables dispositivos para alterar y distorsionar los tonos del oscilador, originalmente sencillos, y para filtrar y modular el resultado. Durante la década de los cincuenta, compositores como Stockhausen, en Alemania, trabajaban afanosamente en pequeños estudios junto a las estaciones de radio locales, produciendo música electrónica "pura". En París, investigando en estrecha relación con ingenieros de sonido de la ORTF (la empresa de radiotelevisión de Francia), Pierre Schaeffer se convirtió en un pionero de lo que él denominó *musique concrète*, música de collage que utilizaba sonidos cotidianos del mundo real.

En Estados Unidos, Bell Telephone Laboratories construyó lo que probablemente haya sido el primer sintetizador. Ocupaba varias habitaciones y su objetivo fundamental era el estudio de la síntesis de la voz humana. La empresa sabía que sus operadores telefónicos, de distintos lugares del país, con frecuencia no comprendían bien los acentos de unos y otros, lo que daba lugar a una elevada proporción de conexiones falsas y números equivocados. Ellos creían, quizá con cierto exceso de optimismo para aquellos tiempos, que una voz sintetizada aceptada a escala universal acabaría con el problema. Numerosos músicos norteamericanos de la actualidad recibieron su formación básica en electrónica en aquel entonces, por cortesía de la empresa Bell.

En Gran Bretaña se estaban realizando trabajos similares aunque a una escala menos ambiciosa. No obstante, el BBC Radiophonic Workshop (taller radiofónico de la BBC) produjo, a principios de los años sesenta, uno de los más grandes clásicos de la música electrónica de todos los tiempos: la banda sonora para la serie de televisión *Doctor Who*.

La primera incursión en el campo de la música por ordenador se produjo ya en 1957, cuando Lejaren Hiller introdujo un conjunto de instrucciones en el ordenador Illiac de la Universidad de Illinois. Estas instrucciones se transformaron en cuatro grupos de datos técnicos que se transcribieron entonces a notación musical. El resultado fue una obra en cuatro movimientos para cuarteto de cuerdas llamada *Illiac suite*. La música en sí, aunque bien adaptada para su ejecución con cello, viola y dos violines, sonaba vaga y como sin rumbo. Sin embargo, no es difícil encontrar otras piezas musicales, producidas de forma convencional por compositores de la época, que suenan aún mucho peor.

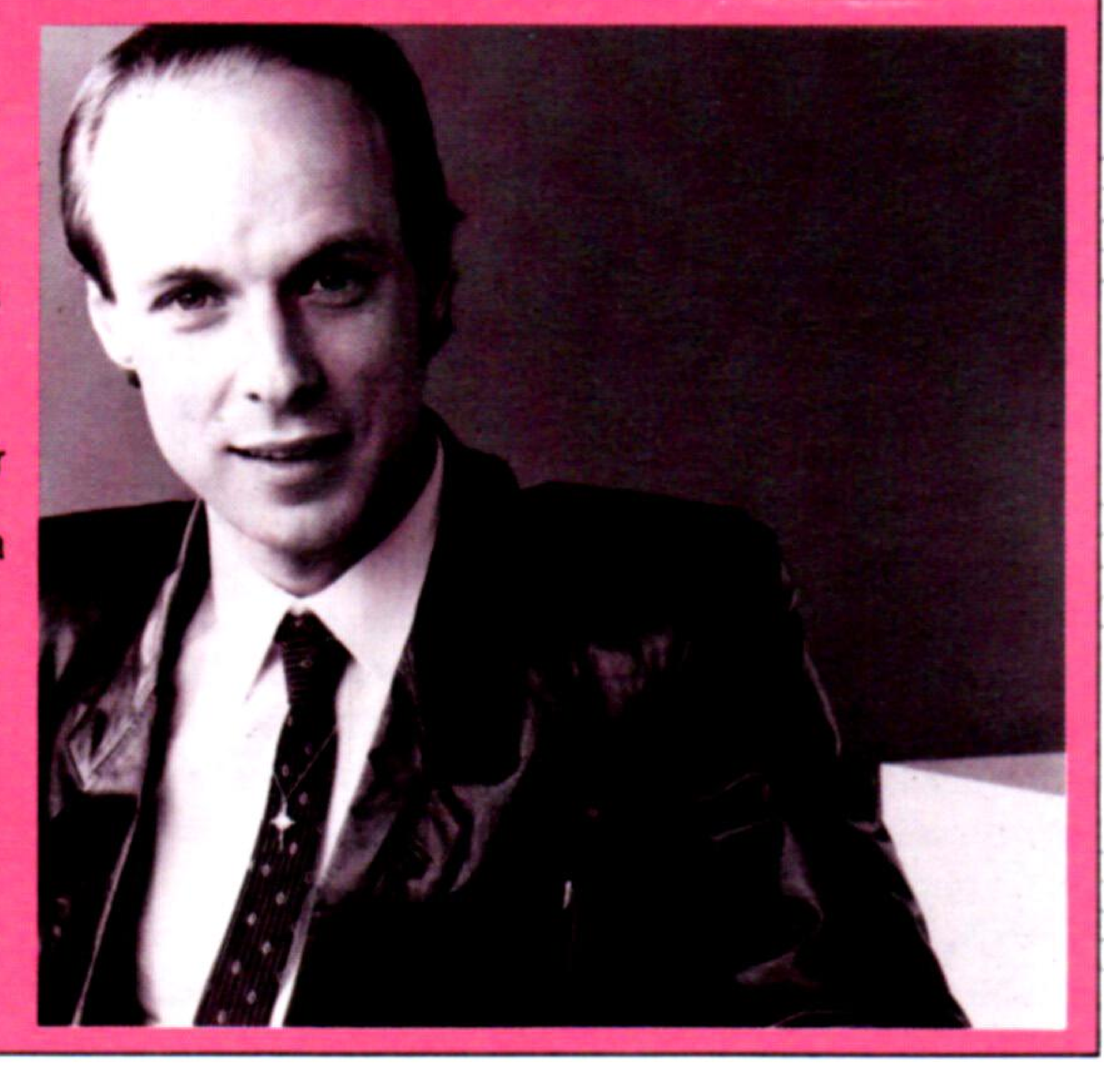
Pocos años después Hiller creó otra obra, esta vez utilizando el ordenador IBM 7090. Diseñó un esquema de programación denominado MUSICOMP (*MUsic SImulator-Interpreter for COMpositional Procedures*: intérprete-simulador musical para procedimientos de composición), que permitió una mayor flexibilidad y variedad en el trabajo para llegar a la composición final. A la misma la bautizó *Computer cantata (Cantata para ordenador)* y está escrita para un vocalista con sonidos electrónicos grabados. Nuevamente, la música resulta interesante a ratos en lugar de ser subyugante. Pero Hiller había demostrado a sus colegas que el ordenador se podía utilizar de forma creativa.

Su trabajo representó sólo una parte de una intensa investigación que se llevó a cabo en las universidades norteamericanas en los años siguientes. John Chowning, otro pionero, empleó más tarde un ordenador para explorar cómo se percibe el sonido mientras la fuente que lo produce se desplaza de un lugar a otro. La utilización de sus trabajos de investigación por parte de Yamaha ha tenido una incidencia directa en el tipo de sintetizador que se está produciendo en la actualidad.

Con la excepción de la música para crear ambientes de ciencia-ficción, la música electrónica permaneció en el ámbito de la música clásica durante varios años y el público tomó conciencia de este cambio de enfoque y de técnica a través de los compositores de vanguardia. En los típicos conciertos de nueva música de los años sesenta participaban varios intérpretes, algunos de ellos tocando instrumentos convencionales, otros dedicados a procesar el sonido de aquellos instrumentos con unidades de separación de frecuencia y filtros. Todos los músicos, incluyendo a los "técnicos", seguían una partitura, pero ésta guardaba poco parecido con la notación musical estándar.

Además de las directrices novedosas, como las que describían las posiciones de los micrófonos y las variaciones de los filtros, los compositores intentaban obtener indicaciones visuales del aspecto que tenían estos nuevos sonidos al ejecutarlos. En algunos casos los músicos tocaban utilizando partituras

Espíritu precursor



Probablemente más conocido por su trabajo con Roxy Music a principios de los años setenta, Brian Eno fue un pionero en la utilización de los primeros sintetizadores. Después de dejar el grupo, en 1973, Eno ha sido uno de los puntales de la música electrónica de vanguardia y música *mood*. También ha colaborado con figuras del calibre de David Bowie y Robert Fripp. Más recientemente, Eno ha trabajado en partituras para televisión y cine y, junto con su hermano, ha desarrollado una partitura para la película de archivo de la NASA sobre aterrizaje lunar



que más bien parecían hojas garabateadas por un diseñador gráfico. Este problema (el de cómo explicar la ejecución de la música, qué lenguaje emplear y cómo visualizar el resultado) persiste en los sistemas de música digital en los años ochenta.

A medida que fue transcurriendo la década de los sesenta, los músicos pop de la floreciente cultura juvenil empezaron a pasarse más tiempo en los estudios de grabación, y también comenzaron a experimentar con música electrónica. El ejemplo clásico es el de los Beatles, quienes hallaron en George Martin no sólo un experto ingeniero de grabación, sino también a un músico que había estado profesionalmente atento a los desarrollos que se habían producido en el campo de la música clásica. Él alentó a los Beatles para que utilizaran todo el estudio como un instrumento musical y al poco tiempo ya estaban aplicando técnicas de collage con cintas e incorporando sonido sintetizado.

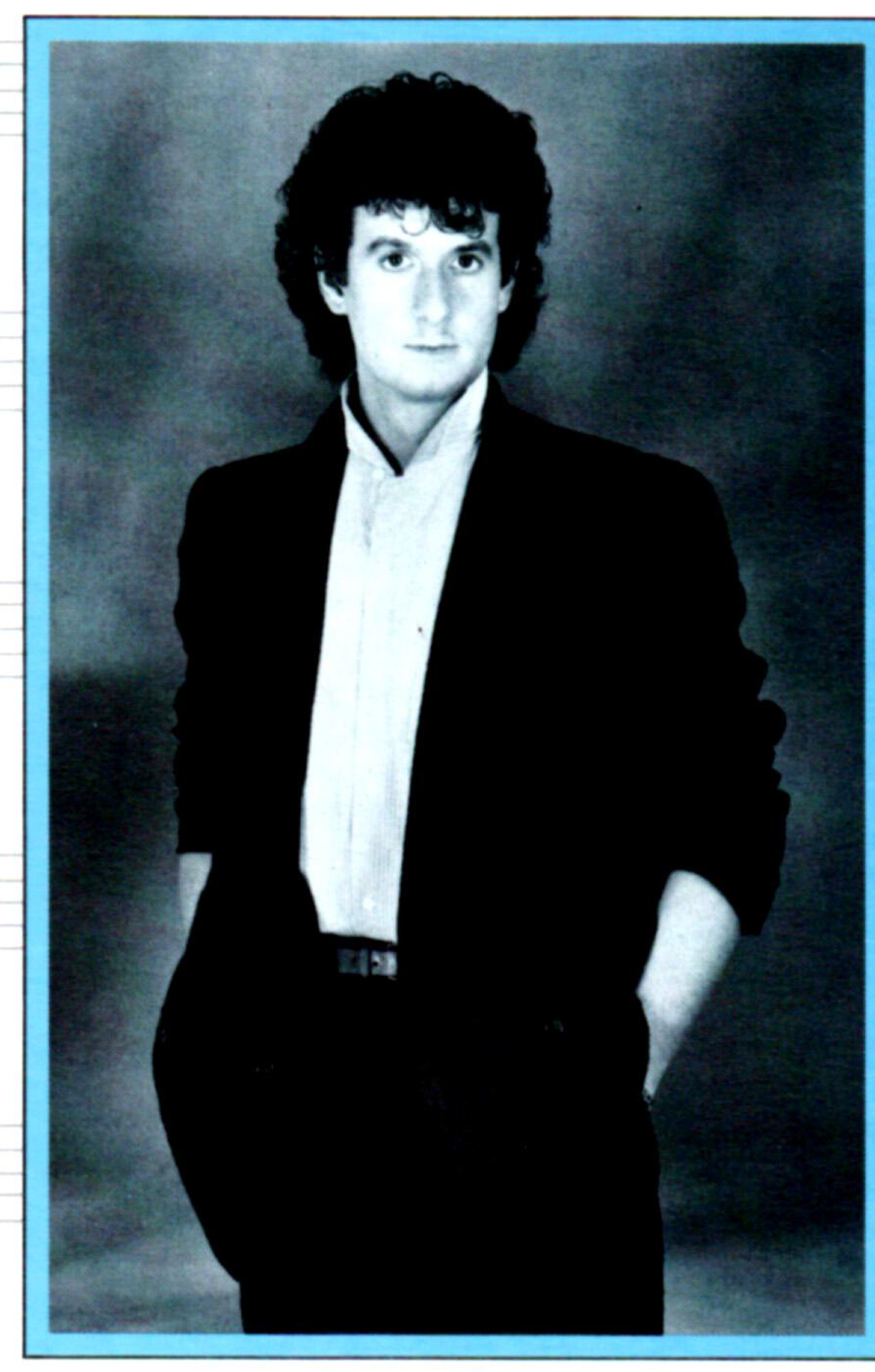
Algunos músicos ganaron su prestigio mediante la utilización de determinadas unidades para proceso de sonido. Un guitarrista como Jimmy Page modeló su estilo en base al de los músicos negros norteamericanos de los años cuarenta, pero utilizando una serie de controles de distorsión produjo un sonido de identificación inmediata como el de Led Zeppelin. Esto, junto al empleo por Jimi Hendrix de mecanismos de autoalimentación de sonido y filtros de barrido rápido (conocidos hoy como "pedales wah-wah"), constituyó la base del *heavy metal*.

En los años setenta las distintas unidades para generación y proceso del sonido, que habían estado disponibles desde los años cincuenta, comenzaron a incorporar diseños transistorizados. Se volvieron más pequeñas, más portátiles y, como resultado, menos restringidas al ámbito de los estudios. Los guitarristas podían tocar en vivo empleando un conjunto de pedales para efectos especiales. Al poco tiempo, los organistas y pianistas tuvieron acceso a sintetizadores de precio asequible que podían utilizar en los escenarios.

Típicamente, estos sintetizadores incluían un juego de osciladores regulables, moldeadores de envolventes (para crear las características del sonido: ataque, sostenimiento y decaimiento), filtros variables, moduladores que podían separar las señales en frecuencias nuevas y generadores de ruido. Así como Jimi Hendrix había sido un modelo para los guitarristas, Brian Eno se convirtió en el modelo para los músicos de sintetizador, principalmente debido a sus vinculaciones con la "nueva música" de la vanguardia clásica.

Al mismo tiempo, los equipos de los estudios de grabación se volvieron más sofisticados, puesto que los músicos buscaban que el estudio les proporcionara algo del proceso de producción que ellos no pudieran crear en el escenario. La mesa de mezclas, ahora diseñada para canalizar grabaciones sucesivas en 16 o 24 pistas de cinta, era todavía demasiado grande como para llevarla de un lado a otro, y la instalación de muchas de las unidades de proceso llevaba su tiempo. En Estados Unidos y en Gran Bretaña surgió una nueva generación de productores. Por lo general habían empezado como ingenieros y estaban mucho más familiarizados con el equipo que los músicos que les habían pagado para que les proporcionaran "el sonido correcto".

En Jamaica los ingenieros empezaron a utilizar la mesa de mezclas como si fuera un instrumento. Las canciones acabadas, grabadas en cinta de pistas múltiples, se volvían a desmontar en sus pistas rítmicas individuales. Las contribuciones originales, vocales o instrumentales, se utilizaban entonces como materia prima para usarlas en la mezcla o eliminarlas de la misma en un estilo que dependía mayormente de la reverberación y las unidades de retardo de señal; éste fue el estilo *dub*.



Productor de ases

Si bien es conocido como productor del grupo británico Culture Club, Steve Levine es más famoso por su habilidad para combinar música electrónica y voces humanas para producir pop *seamless* (bien engranado). Levine fue uno de los primeros productores de Gran Bretaña que hicieron uso del tambor Linn, un sintetizador digital programable, y recientemente ha desarrollado nuevas técnicas de grabación digitales. Levine ha hecho una gran utilización de sintetizadores y otros equipos de música digitales en su reciente single, *Believin'*, que escribió conjuntamente con Boy George, integrante de Culture Club

El advenimiento de los sintetizadores digitales supuso la posibilidad de codificar sonidos no electrónicos. Este proceso se conoce como "muestreo" (*sampling*). Baterías eléctricas como el Linn se convirtieron en objetos muy buscados en los estudios y enseguida se incorporaron a las actuaciones en vivo. A mediados de los ochenta, el muestreo y la manipulación del sonido se han convertido en la "naturaleza del arte", y los estudios y escenarios bien equipados por lo general disponen de más aparatos de equipos digitales que de instrumentos analógicos. Grupos de gran éxito, como el británico Culture Club, combinan sus propias aptitudes musicales para la escritura de canciones con las técnicas digitales de productores como Steve Levine, quien utiliza instrumentos y unidades de proceso que valen decenas de miles de libras.

La necesidad de una interface que conectara a un instrumento con otro, o que ampliara la capacidad de un sintetizador mediante la adición del sistema operativo y la memoria de un microordenador, determinó la unión de los fabricantes de instrumentos musicales. Éstos produjeron las primeras especificaciones del MIDI en abril de 1983 y, desde entonces, pocas empresas se han atrevido a anunciar un nuevo sintetizador que no sea compatible con el MIDI. En el próximo capítulo estudiaremos con detalle los antecedentes y el desarrollo del MIDI.

Errores típicos

La detección y corrección de errores es un aspecto importante del diseño de programas: veamos cómo tratarlos

Existen muchas fuentes de error potenciales en cada una de las etapas de la creación de un programa, desde su especificación, pasando por el diseño y la codificación, hasta la comprobación. Con frecuencia se introducen errores en las etapas de especificación y diseño si se presta poca atención a la naturaleza del problema y si no se tiene el suficiente cuidado para asegurar que el programa haga exactamente lo que se supone debe hacer. Podemos reducir las posibilidades de que se produzcan estos errores siguiendo los métodos de diseño estructurado que esbozamos previamente en el curso (véase p. 956). Es probable que surjan otros errores cuando se traduce el diseño a código —una digitación deficiente puede crear errores (*bugs*), ¡como sabrá cualquiera que alguna vez haya escrito mal el nombre de una variable!— y hasta la comprobación y la depuración pueden dar lugar a otros problemas cuando la corrección de un fallo origina otros.

Pero es en las interfaces (entre rutinas y entre el programa y su usuario) donde se producen la mayoría de los errores. Se debe tener especial cuidado en asegurar que todos los valores que se pasen por estas interfaces sean del tipo de datos correcto y estén comprendidos en el ámbito de valores requerido por el programa. Los valores se pueden verificar dentro de la misma rutina que los pasa o bien en la rutina que los acepta; este proceso de verificar los valores a medida que pasan entre rutinas se conoce por *cortafuegos*.

Para asegurar que los valores que salen de una rutina estén dentro de la escala apropiada y sean del tipo de datos correcto, se debe comprobar si la salida depende de un valor introducido por un usuario o leído de un archivo. Los valores que se introducen en una rutina siempre se deben comprobar. Las subrutinas se pueden diseñar para dar un conjunto de salidas bien definidas, pero los seres humanos no funcionan tan automáticamente y tienden a dar una amplia gama de respuestas diferentes ante cualquier pregunta dada, de modo que en todas las rutinas que acepten datos de operadores humanos se deben colocar comprobaciones rigurosas. De forma similar, los archivos de datos se pueden corromper o leer equivocadamente, de modo que se deben colocar comprobaciones en todas las rutinas de manipulación de archivos.

No es frecuente que los errores hagan que el programa se rompa. Cuando lo hacen es porque el programa ha quebrantado alguna regla de lenguaje (utilizando ilegalmente un operador, como, p.ej., en RESULTADO=PRIMERO$+SEGUNDO$) o una regla del sistema operativo (abrir demasiados archivos a un tiempo, p. ej.). El código que sigue parece que es un programa perfectamente legítimo:

```
10 FOR CONTADOR=1 TO 10
20       SUMA=SUMA+1
30       PRINT CONTADOR,SUMA
40       GOTO 10
50 NEXT CONTADOR
```

Sin embargo, es un algoritmo que no acaba nunca y romperá el programa debido a la forma en que funciona el lenguaje. En este caso, éste (BASIC) utiliza la "pila" para llevar el registro de los bucles FOR...NEXT, sumando a la pila cada vez que se pasa por el principio de bucle (línea 10). En este programa, a la línea 50 (con la instrucción NEXT que disminuiría la pila) no se llega nunca, y, por tanto, la pila se va llenando poco a poco hasta que finalmente se genera un mensaje *stack overflow* (desbordamiento de capacidad de la pila) y el intérprete interrumpe el programa. Por lo general, los errores de este tipo son fáciles de detectar, pero si aparecieran en secciones de código de poco uso sería necesaria una comprobación exhaustiva para descubrirlos.

Un tipo de error más molesto es el que permite que un programa se ejecute normalmente pero dando resultados erróneos. A modo de ejemplo

Lista de comprobación de errores

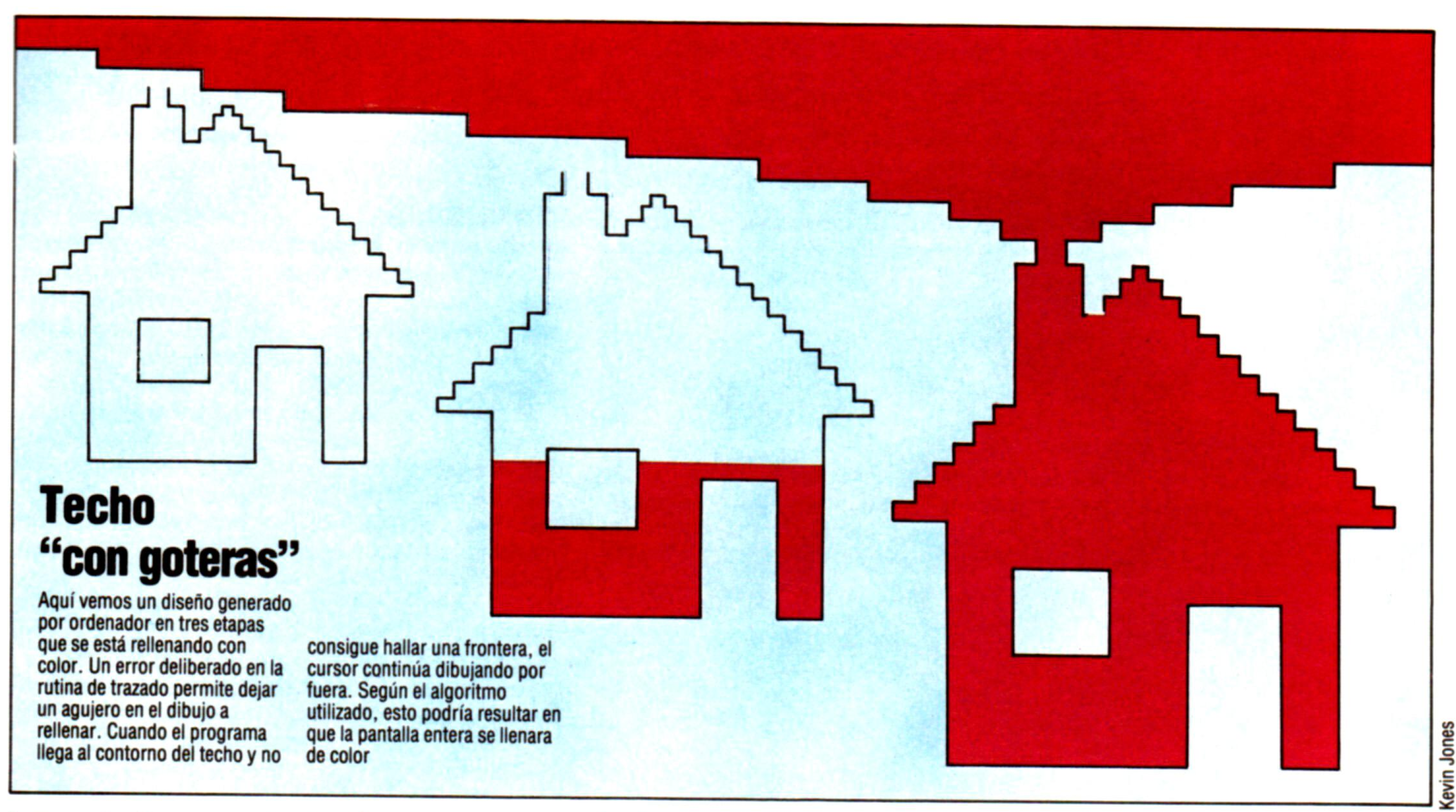
Un enfoque estructurado lógico es lo fundamental para evitar errores y acortar el proceso de depuración; la siguiente lista de comprobación de errores (elaborada a partir de una idea de G.J. Myers en *The art of software testing*) es un ejemplo abreviado de un enfoque de este tipo

Variables
1 ¿Son exclusivos todos los nombres de las variables, teniendo en cuenta que muchos intérpretes utilizan solamente los dos primeros caracteres de cualquier nombre?
2 ¿Se ha vuelto a emplear alguna variable (especialmente los contadores de bucles o los parámetros de subrutinas) mientras su contenido aún era significativo?
3 ¿Están los subíndices de las matrices dentro de los límites, y son números enteros?
4 ¿Empiezan los subíndices de las matrices en el elemento cero o en el elemento uno?

Cálculos
1 ¿Producen los cálculos resultados numéricos o alfanuméricos? Y los resultados, ¿se les asignan a variables numéricas o alfanuméricas?
2 ¿Algún cálculo da como resultado un número demasiado pequeño o grande para que el ordenador lo manipule? ¿Puede esto producir un error de "división por cero"?
3 ¿Pueden ser significativos los errores de redondeo?
4 ¿Están todas las operaciones en una expresión que se ejecute por el orden lógico correcto, en contraposición al impuesto por la precedencia de operadores aritméticos?

Comparaciones
1 ¿Las matrices se comparan siempre con series, y los números con números?
2 ¿Importa que una variable alfanumérica o en serie de verificación esté total o parcialmente en mayúsculas o minúsculas?
3 ¿Se están comparando series de longitud desigual? Y la diferencia en longitud, ¿importa más o menos que las diferencias en caracteres?
4 ¿Se están mezclando adecuadamente operadores booleanos y de comparación? A > B OR C, por ejemplo, no es lo mismo que A > B OR A > C.
5 La precedencia de operadores booleanos y de comparación, ¿afecta a la ejecución de cualquier expresión de comparación?

Control
1 Los bucles y algoritmos, ¿terminan sea cual sea el estado de las variables?
2 ¿Tienen los bucles y las rutinas un único punto de entrada y salida cada uno?
3 Cuando fracasa una sentencia IF...THEN, ¿el control pasa a la siguiente sentencia del programa o a la siguiente línea del programa?
4 ¿Qué sucede si no se satisface ninguna de las condiciones de una sentencia de bifurcación múltiple?



Techo "con goteras"

Aquí vemos un diseño generado por ordenador en tres etapas que se está rellenando con color. Un error deliberado en la rutina de trazado permite dejar un agujero en el dibujo a rellenar. Cuando el programa llega al contorno del techo y no consigue hallar una frontera, el cursor continúa dibujando por fuera. Según el algoritmo utilizado, esto podría resultar en que la pantalla entera se llenara de color

Kevin Jones

hemos elegido un patrón de relleno que traza un dibujo en la pantalla y después lo rellena con color. Las rutinas de relleno buscan las fronteras del dibujo. Cuando se alcanza una frontera, el ordenador hace dar la vuelta al cursor y continúa dibujando hasta alcanzar otra frontera. Para que una rutina de relleno funcione, las fronteras deben estar bien definidas y completas. Dicho en otras palabras, en el esquema del dibujo no puede haber espacios abiertos ya que, de lo contrario, la rutina de relleno desparramaría el color por fuera de las fronteras.

Las versiones de BASIC que utilizan la mayoría de los micros personales hacen que el tratamiento de errores resulte relativamente sencillo, produciendo mensajes de error claros y concisos y permitiendo que un programa roto continúe después de haber modificado los valores de las variables por el teclado, facilidad muy útil cuando se está depurando un programa. La mayoría de las versiones del BASIC permitirán el empleo de una instrucción como ON ERROR GOTO para transferir el flujo de control a una rutina especial para tratamiento de errores y controlar así errores que de otra forma serían decisivos. Se realiza incluyendo una línea como ésta:

```
30 ON ERROR GOTO 20000: REM rutinas para
   tratamiento de errores
```

cerca del comienzo del programa. Cualquier error hará entonces que el programa actúe como si se hubiera encontrado con la instrucción GOTO 20000. Por lo general ON ERROR también modifica dos variables; la primera de ellas almacena un código de error que indica la clase de error que se ha producido, y la otra simplemente retiene el número de línea en la cual se ha producido el error. Los nombres otorgados a estas variables y los códigos de error resultantes variarán de una máquina a otra, de modo que debe consultarse el manual. Al producirse un error, el flujo del programa se desvía hacia la línea 20000, ahí se identifica el error a partir del número hallado en la variable correspondiente y se emprende la acción apropiada.

Un programa bien escrito no tendrá más de una rutina ON ERROR. Dicha rutina no será capaz de tratar los errores de sintaxis, agotamiento de la memoria, desbordamiento de capacidad (*overflow*) de la pila, etc. Lo mejor que puede ofrecer esta facilidad es una ordenada suspensión del trabajo por parte del sistema, asegurando que todos los archivos queden cerrados (CLOSE) y que el usuario sepa exactamente qué es lo que ha sucedido.

Algunos errores, como por ejemplo una división por cero, que una rutina de este tipo podría manejar, en realidad se deben tratar de una forma diferente. Existen varias razones para ello:

- La instrucción ON ERROR GOTO y el subsiguiente salto hacia atrás al programa principal constituye una entrada y una salida extras hacia y desde una rutina. Ello viola el principio de la programación estructurada que establece que las rutinas han de tener un solo punto de entrada y un solo punto de salida.
- El lugar correcto para protegerse contra una división por cero es la propia rutina que realiza la división. Es una mala costumbre diseñar algoritmos que puedan romper el sistema. Si la verificación extra de errores implicada ralentiza el programa hasta un nivel inaceptable, se debe volver a diseñar la rutina de modo que no exista este riesgo.
- Las rutinas para tratamiento de errores se complican muy rápidamente si IF...THEN...ELSE enlaza con múltiples salidas. Se ven inevitablemente limitadas por la numeración de líneas del resto del programa y, por consiguiente, se deben reescribir siempre que se vuelva a diseñar cualquier rutina que las utilice. Éstas son particularmente difíciles de diseñar, comprobar y depurar, y cualquier error que exista en una rutina de esta clase puede introducir problemas de mayor envergadura al desviar el flujo de control de formas imprevistas.

Rompiendo barreras

El acceso ilícito a los ordenadores centrales utilizando máquinas personales y modems se conoce como "hacking" (asalto)

La película *Juegos de guerra* cautivó la imaginación de muchos usuarios de ordenadores personales. Utilizando un micro y un modem, el héroe se comunica ilícitamente con una sucesión de ordenadores para modificar los resultados de sus exámenes escolares, reservar billetes en líneas aéreas y cargar el software para los juegos más novedosos. Sin embargo, las cosas comienzan a ir mal cuando inadvertidamente consigue acceso al ordenador NORAD, responsable de la defensa aérea norteamericana, y está a punto de desencadenarse una guerra nuclear mundial. Una bonita historia para divertirse; ¿pero seguro que no es demasiado rebuscada?

Muchas "interrupciones" por ordenador de este tipo se han producido realmente, y el causante ha resultado ser, por lo general, un adolescente provisto de un micro personal y un modem. Las "víctimas" van desde potentes ordenadores centrales pertenecientes a universidades y grandes corporaciones, hasta servicios de tablones de comunicaciones dirigidos por entusiastas en microordenadores. Todos los ordenadores que permiten el acceso externo mediante el teléfono son vulnerables. En 1983, la realidad estuvo muy cerca de imitar a la ficción cuando se sospechó que dos "asaltantes" habían conseguido acceder al sistema de ordenador NORAD de Omaha (Nebraska).

Distribución ilegal



A pesar de que la empresa norteamericana Pepsi-Cola afirma no tener conocimiento del incidente, la misma fue víctima de un caso de "asalto" muy celebrado en los últimos años. De acuerdo a los informes, alguien en Estados Unidos accedió de forma ilegal a un ordenador de la empresa Pepsi-Cola en Canadá. Los "asaltantes" enviaron grandes partidas de Pepsi a puntos de destino preelegidos, con el objeto de desviar elevadas sumas de dinero hacia cuentas corrientes ilícitas

Ian McKinnell

Los dos adolescentes implicados eran de Los Ángeles y se las habían apañado para meterse en el Arpanet (la red secreta de ordenadores que posee el Departamento de Defensa de Estados Unidos). Utilizando un Commodore Vic-20 y un Tandy TRS-80, la pareja se las arregló para explorar el contenido de varios ordenadores conectados con el Arpanet, que suelen pertenecer a contratistas de defensa, organizaciones de investigación y universidades. A pesar de que no lograron obtener información secreta (el sistema se emplea básicamente para reunir datos científicos), la facilidad con que los dos adolescentes se "colaron" causó un enorme desconcierto en el Departamento de Defensa.

El motivo por el cual a los muchachos les resultó tan fácil tuvo más que ver con la indolencia humana que con cualquier fallo del ordenador. Todos los usuarios registrados del Arpanet poseen contraseñas; lamentablemente, éstas no se eligieron con mucha imaginación. En este caso, los muchachos supusieron que la Universidad de California de Berkeley podría ser usuario del Arpanet. A partir de este acierto, la contraseña "UCB" los introdujo en la red y entonces se vieron en libertad para acceder a cualquiera de los ordenadores conectados al Arpanet, uno de los cuales es el NORAD, de las oficinas centrales subterráneas de Omaha.

A pesar de que la sede central del NORAD está en el Arpanet, los ordenadores responsables de la verdadera defensa aérea no se encuentran allí. Están situados debajo de las montañas Cheyenne, en Colorado, y no están conectados a las líneas telefónicas públicas.

Puede que los ordenadores NORAD sean inmunes a la violación por parte de "asaltantes", pero otros no lo son. Otro incidente se produjo en julio de 1983, cuando un grupo de adolescentes de Milwaukee entró en más de 60 ordenadores pertenecientes a facultades, corporaciones y al Los Alamos National Laboratory, que se dedica a la fabricación de armas. Nuevamente, según las autoridades no se accedió a información secreta, sólo a registros, informes de rutina y mensajes. Se llamó al FBI para que averiguara por qué el grupo, cuyos miembros se autodenominaban los "414", había podido realizar tamaña proeza. Los 414 manifestaron que en ninguno de los ordenadores a los que habían llamado había mensajes de seguridad.

Éstos son apenas algunos de los casos a los que se ha dado publicidad. Muchos otros no llegaron a trascender a la opinión pública, porque son muy pocas las organizaciones que quieren que se sepa que en su ordenador central se ha infiltrado, supongamos, un joven de 17 años con un micro personal que vale 25 000 pesetas. Asimismo, muchas organizaciones no están al tanto de las infiltraciones: suele ser muy difícil saber si un usuario no registrado o impostor ha estado "en línea", aunque algunos de los "asaltantes" más descarados dejan mensajes desafiando a que los cojan y firman como "El aniquilador del sistema" o "El capitán Zap".

¿Cómo se realiza exactamente el *hacking* (asalto)? Todo "asaltante" potencial necesita un ordenador personal, un modem y una pizca de ingenio. El primer obstáculo es averiguar el número de teléfono de un ordenador. Para las redes de acceso público, como la Telecom Gold, de Gran Bretaña, o The Source, de Estados Unidos, ello no es difícil, ya que por lo general se anuncian profusamente.

Ensayo y error



Ian McKinnell

El procedimiento que utiliza un "asaltante" supone un gran trabajo de ensayo y error que en muy contadas ocasiones conduce a lograr introducirse en el sistema sin autorización. Incluso aunque un "asaltante" sea capaz de localizar un sistema determinado, a menudo se encontrará con un diálogo de esta naturaleza cuando intente colarse en el sistema:

CONEXION QX001001 14, 32 12/7/84	
>	**Pulsar return: se solicita ID del usuario**
USER?	
> HELP	**No hay ayuda**
USER?	
> $ OFF	**Primer intento de ID válida**
USER?	**ID no válida para acceso**
> BN001	**Segundo intento de ID válida**
PASSWORD?	**ID aceptada: se solicita la contraseña**
> HUESPED	**Primer intento**
PASSWORD?	**No se acepta: 2.ª solicitud**
> NUEVOUSUARIO	**Segundo intento**
PASSWORD?	**No se acepta: 3.ª solicitud**
> QWERTY	**Intento desesperado**
LOGOFF 15.13 CONNECT TIME = 0.13 MINS	**El usuario queda desconectado después del tercer intento fallido por descubrir contraseña**

En algunos casos, ya sea por elucubración o por pura suerte, una persona logra descubrir una contraseña de gran prioridad válida y se mete en el sistema. El siguiente diálogo imaginario describe cómo un usuario de esta clase podría acceder a información confidencial relativa a un usuario legal:

CONNECT BYF990 15.14.02 12/07/84	
>	**Pulsar retorno**
USER?	
> BN001	
PASSWORD?	**ID válida: solicita contraseña**
> SYSOP	**Primer intento: operador sistema**
LOGON 15.15.07 12/07/84	
HOST: BYF990/SPYLOM	
USER: BN001	
SERIAL NO: ZA180-7	**Número de serie del software**
PRIORITY: SUPERUSUARIO	**Paso libre a los archivos**
STATUS: ACTIVE	**Los usuarios se pueden considerar inactivos si no usan el sistema regularmente**
YOU ARE SYSOP	**Confirmación**
7 USER(S)	
APP01 APP02 BYF7 BTY04 BZX88 BZX02 SYSOP	**Lista todos los usuarios**
> REMOVE ARP01	**Instrucción reservada para operador del sistema**
USER(S) ARP01 DISCONNECTED(S)	**Usuario válido eliminado**
> WHO IS BTY04	
BTY04 ROSA RUPEREZ, BTY LOPP, 742 SILICON DV HERTS 07662-093164	

En el caso de los ordenadores privados ya es un poco más difícil. Pero si se sabe aproximadamente dónde está situado el ordenador, entonces se puede utilizar la técnica empleada por el protagonista de *Juegos de guerra*: él programó a su micro para que marcara todos los números de teléfono posibles de su ciudad. Si respondía un ordenador (identificado por el silbido delator del tono de transmisión) entonces la máquina tomaba nota del número; pero si respondía una persona, el modem colgaba y marcaba otro número. Esto se puede realizar con un modem de llamada automática; ¡marcar los números a mano resultaría bastante pesado!

Una vez conectado con el ordenador, invariablemente se le solicita una contraseña. Algunas redes permiten un uso limitado si se digita "HUESPED" o "USUARIONUEVO", o si simplemente pulsa RETURN. Pero un verdadero "asaltante" probará y romperá la contraseña. Con frecuencia esto no es especialmente difícil, porque los usuarios tienden a hacer poco alarde de imaginación y utilizan nombres, como "SMITH", o palabras obvias como "SECRETO", o incluso simplemente "CONTRASEÑA". Del mismo modo, en el caso de las contraseñas compuestas por números, la gente tiende a escoger secuencias fáciles de recordar: por ejemplo, su fecha de nacimiento, como "090560". Muchos ordenadores son muy indulgentes y permiten varios intentos de contraseña antes de desconectarlo a uno. Aun así, normalmente se puede volver a marcar y seguir desde donde se quedó antes sin producir sospechas en el ordenador anfitrión.

Una vez en el sistema, la mayoría de los "asaltantes" se conforman con mirar los archivos, encontrar las páginas de juegos (si las hubiera) y "hablar" con otros "asaltantes" que también hayan conseguido meterse. Algunos de los más destructivos eliminan archivos, dejan mensajes obscenos e intentan "romper" todo el sistema; esto último puede tener consecuencias desastrosas para los usuarios.

Aun en los ordenadores centrales más sofisticados, es frecuente que los programadores dejen "puertas secretas" en el sistema de modo que, en un caso de emergencia, puedan evitar las medidas de protección e introducirse en el programa. Muchas veces las personas que operan el sistema desconocen la existencia de puertas de este tipo.

Usted habrá observado que la mayoría de los casos que hemos reseñado se refieren a ordenadores de universidades. Ello se debe a que esta clase de ordenadores, aparte de tener un acceso por línea exterior, por lo general operan con una política de acceso libre. Con miles de usuarios y muchos lugares remotos, ésta es la forma más práctica de llevar un sistema de este tipo. Lamentablemente, también son presa fácil para los "asaltantes" que deseen entrar en ellos, y una vez allí pueden "ir saltando" de un ordenador a otro actuando como si fueran usuarios legítimos. Un estudiante del *campus* de San Jose State descubrió un agujero en un bucle del programa *Talk* del ordenador de la universidad, que permite que los estudiantes "hablen" con otras ciudades universitarias de la California State University. El estudiante consiguió superar la restricción local y logró comunicarse con ordenadores de Suecia, Irán y China, así como de diversos lugares de Estados Unidos. La factura del teléfono ascendió a más de $ 10 000.

¿Qué persiguen los "asaltantes" al actuar de esta manera? Muchos de ellos poseen un código de conducta extraoficial y no borran archivos ni dejan mensajes obscenos. Para ellos la emoción reside simplemente en romper los códigos.

Durante mucho tiempo los bancos han sido objeto de delitos por ordenador, pero hasta hace poco todos ellos se hacían "de puertas adentro", es decir, por poner un ejemplo, empleados deshonestos que transferían dinero a cuentas falsas. Las estimaciones sobre el fraude por ordenador varían, sólo en Gran Bretaña, desde £30 millones hasta más de £2 500 millones al año. Como es fácilmente comprensible, es poco frecuente que los bancos y las empresas admitan con carácter público que han sido víctimas de un fraude por ordenador y, por tanto, es difícil efectuar estimaciones exactas.

El incremento en el número de propietarios de ordenadores personales y la creciente cantidad de redes de ordenadores que utilizan las líneas telefónicas significan que los delitos por ordenador sólo pueden seguir una espiral ascendente.

Acumuladores

Profundicemos un poco en la utilización de acumuladores mediante un ejemplo práctico

Los clientes de una caja de ahorros o de un banco se agrupan en tres categorías según su tipo de cuenta: CA, CB y CC, y a cada una de estas categorías se aplican intereses diferentes, del 10, 12 y 15 %, respectivamente. Para cada uno de los clientes deben calcularse los intereses correspondientes a su saldo y visualizar la nueva cantidad una vez incrementada con los citados intereses. De igual manera, cuando ya se haya procesado la totalidad de clientes, se imprimirá el total por categoría de los intereses entregados.

Con lo aprendido hasta el momento, se conseguiría la representación de la figura 1, ante la imposibilidad de poder agrupar en un punto determinado de la secuencia las operaciones repetitivas. En caso de no hacerse así, no se podría después diferenciar en qué registro deben acumularse los intereses como muestra la figura 2. Al llegar al punto en el que figura el interrogante, el proceso no continuaría por falta de datos, o si lo hiciera sería por defecto, en cuyo caso el resultado sería erróneamente dirigido siempre al mismo acumulador.

Figura 1

Figura 2



Un nuevo hito

Este micro lanzado recientemente es la primera máquina personal que proporciona una unidad de disco incorporada

La etiqueta del precio del Einstein indica que gustará principalmente al usuario personal "serio". Claro está que se puede utilizar para juegos, pero en este campo ofrece pocas ventajas en relación a máquinas que valen la cuarta parte que ésta. Su competidor más cercano, en cuanto a precio y rendimiento, es el BBC Micro, pero los 80 Kbytes de RAM del Einstein le sacan ventaja a los más bien escasos 32 Kbytes que ofrece Acorn.

La unidad de disco está montada en un panel justo arriba del teclado en la carcasa del Einstein, que es inusualmente grande. Esta carcasa tiene la solidez suficiente como para aguantar el peso de una pantalla o un televisor, de manera que el sistema completo no ocupa mucho espacio.

La principal ventaja de la unidad de disco integral probablemente sea la disponibilidad de software. Si bien existen otras máquinas personales, como el Commodore 64, que se podrían equipar con unidades de disco, el hecho de que hayan sido diseñadas pensando en grabadoras de cassette significa que la mayor parte del software se fabrica en cassette y que, en consecuencia, los propietarios de unidades de disco no podrán aprovechar al máximo el superior soporte de almacenamiento de que disponen. La unidad incorporada del Einstein asegura que todo el software se suministrará en disco desde el principio. El empleo de discos permite cargar programas y datos rápidamente y posibilita la utilización de archivos de acceso directo en vez de los archivos de acceso secuencial a los que se ven limitadas las máquinas basadas en cassette. En cada cara del disco de 3 pulgadas se pueden almacenar 190 Kbytes de datos, pero el Einstein sólo se puede utilizar una cara a la vez. En la carcasa se puede instalar una segunda unidad de disco, y en una interface situada en la parte posterior de la máquina se pueden enchufar otras dos unidades.

Para controlar el empleo de la unidad de disco, el Einstein posee su propio sistema operativo de disco (DOS: *disk operating system*). Éste posee muchas similitudes con el CP/M estándar que utilizan muchas máquinas de oficina, y Tatung confía en que las casas de software convertirán los programas CP/M para su ejecución en el Einstein. El sistema operativo y el BASIC Einstein no están contenidos en ROM, como ocurre generalmente, sino que se deben cargar desde disco cada vez que se conecta la máquina. Ello ofrece dos ventajas fundamentales: como el BASIC se carga sólo cuando se lo necesita, otros lenguajes de programación o programas en lenguaje máquina pueden utilizar el espacio completo de RAM, y tanto el sistema operativo como el BASIC se pueden actualizar fácilmente mediante la adquisición de un disco que contenga las nuevas versiones. Tatung tiene planeado ofrecer el lenguaje DR LOGO en el disco que se suministra de forma gratuita con la máquina; pero este obsequio no es todo lo generoso que puede parecer, porque el manual de LOGO se venderá aparte. Una vez cargado el BASIC desde el disco, el Einstein posee unos valiosos 43 Kbytes de RAM disponibles para el usuario, que es más de lo que ofrece cualquier otra máquina personal. Esto es factible debido a que la RAM del Einstein contiene 16 Kbytes de memoria separados que se controlan mediante el chip de gráficos y se usan para la visualización en pantalla.

El BASIC Einstein parece ser una mezcla de BASIC BBC y Microsoft Extended BASIC (como el que utilizan las máquinas MSX japonesas). Incluye instrucciones para renumerar programas y para producir números de línea de forma automática, lo que facilita la entrada de programas. Un editor de pantalla completo ofrece la posibilidad de efectuar cambios en cualquier lugar de la visualización en pantalla. Las instrucciones para gráficos permiten dibujar líneas, círculos y elipses, así como rellenar



Sistema serio
El ordenador Einstein, fabricado por Tatung (antiguamente Decca). La máquina está destinada al usuario personal serio y viene equipada con una unidad de disco. Es más grande que la mayoría de las máquinas personales, lo que permite apoyar la pantalla sobre la carcasa

Chris Stevens

formas con color sólido. Los gráficos tienen una resolución máxima de 256 × 192 pixels y hay 15 colores disponibles, si bien no se pueden utilizar más de dos para cada fila de ocho pixels. El usuario puede definir hasta 32 caracteres de sprites; las instrucciones para controlarlos están incluidas en el BASIC, lo que permite escribir excelentes programas para juegos de acción muy rápida. Se suministra un programa monitor en ROM para hacer que la programación en lenguaje máquina resulte más sencilla.

Asimismo, el chip de gráficos limita la visualización a 40 caracteres a lo ancho. Existen planes para poner en el mercado un accesorio que permita reproducir en el Einstein la visualización de 80 caracteres que exigen muchos programas CP/M; la pantalla de 80 columnas será sólo monocromática, pero hay una versión en color prevista para 1985.

La calidad de sonido del Einstein es buena, con la salida dirigida a un gran altavoz situado encima del teclado. Se proporciona un control de volumen; las instrucciones del BASIC para generar sonido son amplias y fáciles de emplear.

Existen ocho teclas de función; éstas se pueden programar para producir palabras e instrucciones comúnmente utilizadas, y una banda plástica transparente permite fijar etiquetas encima de cada tecla, al estilo del BBC. El teclado está bien construido y la mecanografía al tacto no debería suponer ningún problema, pero Tatung sólo ha suministrado dos teclas de cursor en vez de las cuatro habituales. Esto significa que las teclas del cursor se deben utilizar junto con la tecla de cambio para producir el movimiento en dos de las cuatro direcciones, lo que resulta incomprensible en una máquina tan cara. Se puede producir un juego de caracteres para gráficos desde el teclado si se mantiene pulsada la tecla para gráficos, pero los mismos tienen un empleo limitado.

En cuanto a cantidad de interfaces disponibles, el único competidor del Einstein es el BBC Micro. Éstas incluyen una interface Centronics estándar para conexión con impresora; un conector RS232 para utilizar con impresoras, modems y otros accesorios; un conector para una pantalla a color RGB; una salida para visualización en televisor y un par de conectores para palanca. A los conectores para palanca también se les puede dar una utilidad más seria, dado que son del tipo convertidor de analógico a digital que permite medir voltajes eléctricos. Los dos conectores proporcionan cuatro canales de A a D; éstos se complementan con una puerta para el usuario de ocho bits que puede recibir y producir señales digitales desde y hacia otros componentes del equipo. Esta combinación de puertas para el usuario y de A a D hace que el Einstein sea ideal para control en robótica y aplicaciones científicas.

La futura ampliación es factible en función del "Pipe" (similar en concepto al "Tube" del BBC Micro), que permitirá instalar diversos accesorios. Un conector de ROM dentro de la máquina permite ampliar a 32 Kbytes los ocho Kbytes de ROM que tiene como estándar.

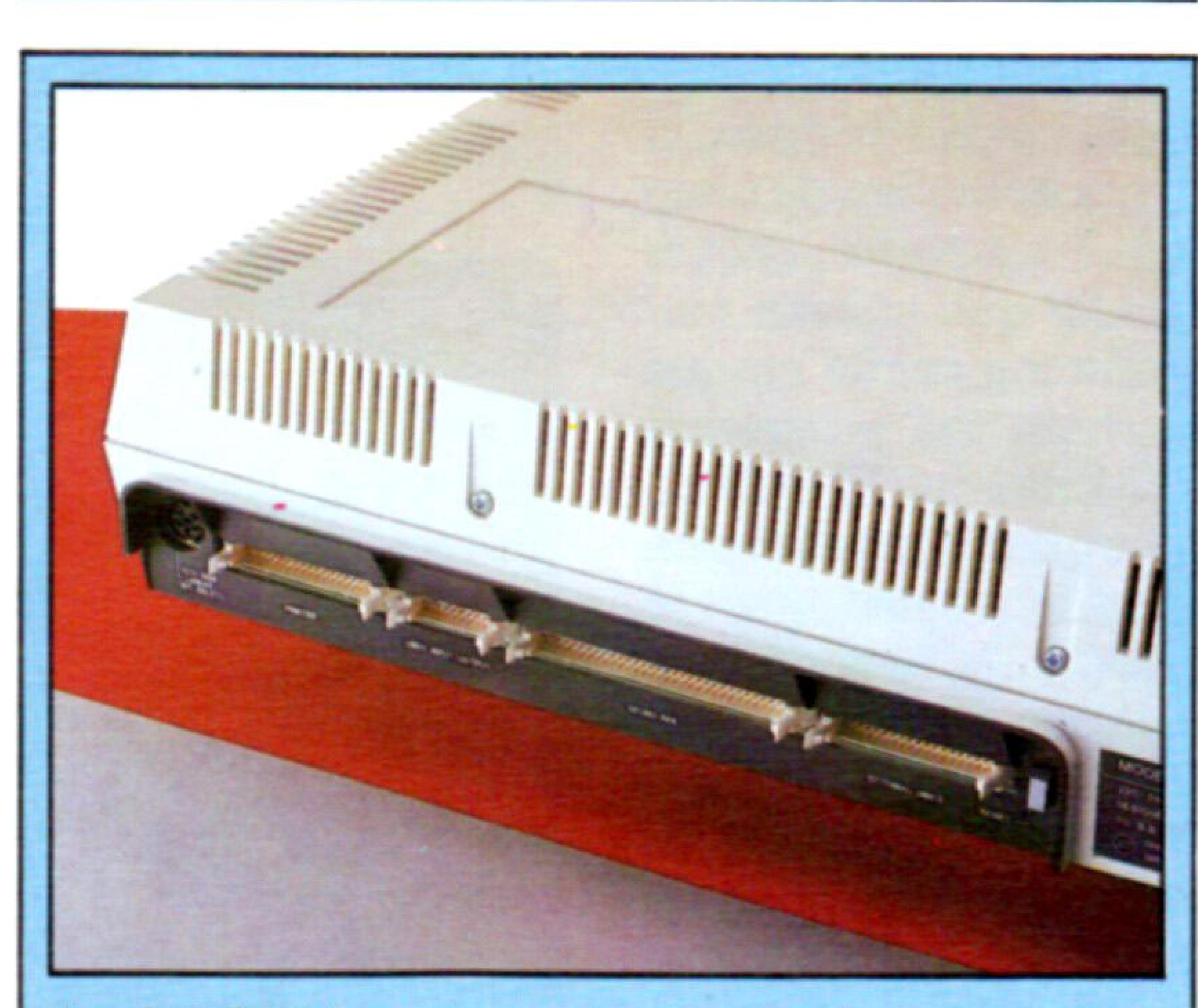
El Einstein, fabricado en Gran Bretaña, sin duda alguna bien vale su precio; pero la realidad es que son pocos los usuarios que se pueden permitir gastarse el equivalente de £500 en un ordenador personal. Todavía es muy escaso el software disponible, y la situación no cambiará a menos que la máquina alcance un volumen de ventas considerable.



Opciones de pantalla
La pantalla Tatung acepta entradas RGB o YUV; ésta última es el sistema propio de Tatung, supuestamente superior. Dado que la salida YUV incluye una línea de video compuesto, con ella se puede utilizar cualquier pantalla



Unidad de disco Hitachi
La unidad de disco responde al sistema Hitachi de 3 pulgadas, que se está haciendo cada vez más popular entre los microordenadores. Los discos pueden almacenar hasta 190 Kbytes, y si bien la unidad sólo lee una cara, los discos se pueden dar vuelta de forma manual para utilizar las dos caras



Interface Einstein
El Einstein está bien dotado de interfaces, incluyendo el "Pipe" Tatung, una puerta de propósito general. También hay una interface para unidades de disco adicionales

Microprocesador Z80

8 K de ROM
Este chip contiene el programa monitor de código máquina. El conector vacío adyacente es para ampliación

Fuente de alimentación conmutada
No utiliza un transformador de potencia convencional y, por lo tanto, es de menor tamaño y tiene un mayor rendimiento, disipando menos calor

Unidad de disco
Formato Hitachi de 3 pulgadas, por una sola cara; 190 K de capacidad

Teclas de función
Definibles por el usuario en modalidad directa



Conectores de unidad de disco

Puerta Pipe para ampliación
Similar al Tube del BBC Micro, permite la conexión de accesorios tales como un segundo procesador

Puerta para el usuario de 8 bits

Interface Centronics para impresora

Salida RGB para pantalla en color

64 K de RAM

Interface RS232

Dos conectores para palanca
Configuran cuatro canales de analógico a digital (A a D)

Control de volumen del altavoz

Modulador de RF
Produce la señal de TV

Segunda unidad de disco
Aquí hay espacio reservado para una segunda unidad opcional

Altavoz

TATUNG EINSTEIN

DIMENSIONES
510 × 430 × 105 mm

CPU
Z80

MEMORIA
80 K de RAM, incluyendo 16 K de memoria de pantalla. Ocho K de ROM, ampliables a 32 K

PANTALLA
Texto: 32 columnas × 24 filas/40 columnas × 24 filas. Gráficos: 256 × 192 pixels con 15 colores

INTERFACES
Centronics, RS232, dos palancas de mando (de A a D), RGB, "Pipe", puerta para el usuario y unidad de disco externa

LENGUAJES DISPONIBLES
Se proporcionan BASIC y DR LOGO. Se venden PASCAL y FORTH

TECLADO
67 teclas tipo máquina de escribir, trazado QWERTY. Incluye ocho teclas de función programables

DOCUMENTACION
Tres manuales, incluyendo uno de BASIC pobre

VENTAJAS
Unidad de disco incorporada, muchas interfaces, enorme memoria para el usuario, buen BASIC, gráficos sprite

DESVENTAJAS
Poco software disponible. Precio de venta al público elevado

Chris Stevens

Explosión final

Concluimos nuestro juego con un listado completo del programa y ofrecemos las líneas alternativas para ejecutarlo en el Electron

La modalidad de gráficos 7 del BBC Micro, también conocida como modalidad de teletexto, posee varias características de las que no dispone en ninguna otra modalidad. Éstas se utilizan para la visualización de información transmitida desde fuentes exteriores, como el Micronet, al que se puede acceder utilizando el ordenador y una línea telefónica normal. Las características para gráficos adicionales que se pueden obtener mediante el empleo de la modalidad 7 permiten producir atractivas visualizaciones con letras mediante unas sencillas y breves instrucciones; por consiguiente, esta modalidad es una alternativa ideal para nuestra pantalla de "final del juego".

Mediante la utilización de códigos de control CHR$ en sentencias PRINT podemos controlar los colores del texto y del fondo, crear texto "intermitente" y producir caracteres de doble altura. Podemos emplear la función TAB de la forma normal para posicionar el texto en la pantalla de 40 por 25 caracteres. Hay siete colores disponibles y éstos se pueden seleccionar mediante los siguientes códigos de control:

129	rojo
130	verde
131	amarillo
132	azul
133	magenta
134	cyan
135	blanco

Siempre que se emplea la modalidad 7, el texto se visualiza en blanco sobre fondo negro. El color del texto se puede modificar en cualquier punto de una sentencia PRINT mediante la inserción de un código de control. Por ejemplo, las tres palabras de la frase de la siguiente línea se imprimirán en rojo, blanco y azul, respectivamente:

```
PRINT CHR$(129)"JUEGO";CHR$(135);"SIN";
      CHR$(132);"FRONTERAS"
```

Es importante, sin embargo, comprender que cuando se imprima otra línea se restaurará el color por defecto (blanco). Por lo tanto debemos utilizar códigos de control en cada nueva línea, aun cuando deseemos seguir imprimiendo en el mismo color.

Además del color del texto, también podemos seleccionar los colores de fondo. CHR$(157), seguido del color que deseamos para el fondo, nos permite hacer esto. Por ejemplo, para producir letras azules sobre fondo blanco se utiliza la siguiente combinación de códigos de control:

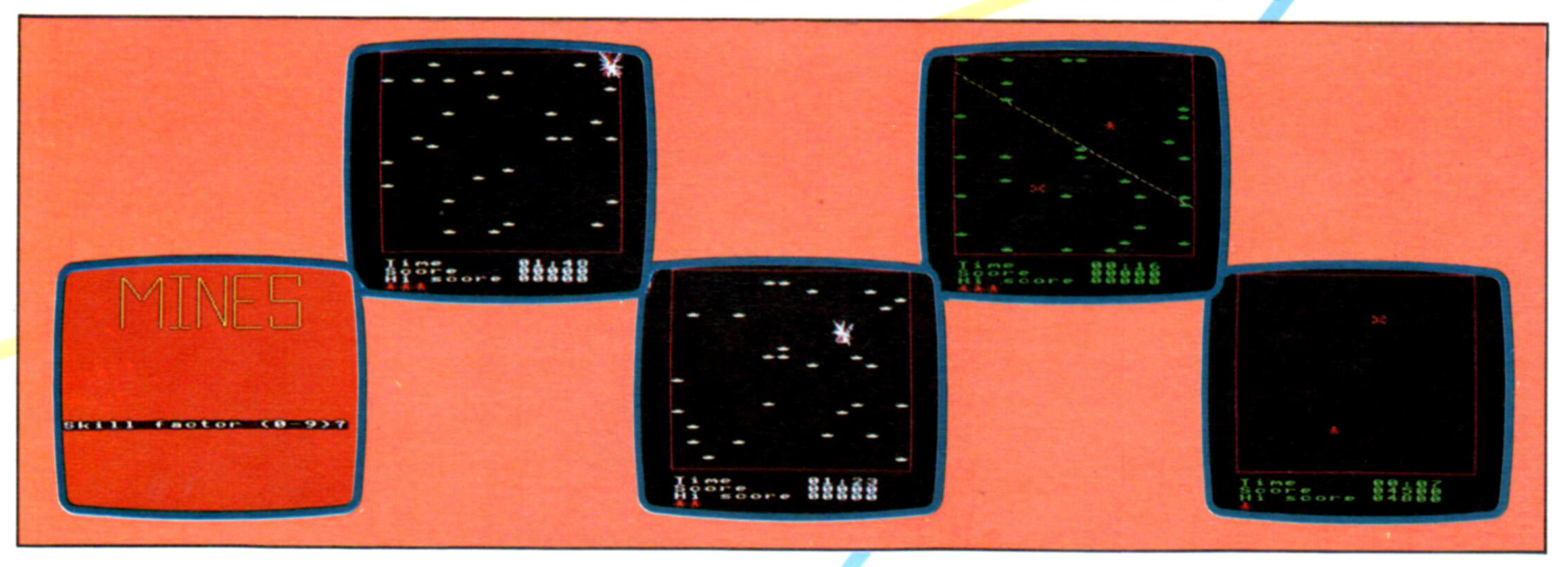
```
PRINT CHR$(132);CHR$(157);CHR$(135);"EN LA
      MARINA"
```

El primer código de control especifica el color del texto; el segundo y el tercero definen el fondo. Se puede hacer que tanto el texto como el fondo sean intermitentes utilizando un código de control CHR$(136) inmediatamente antes del código de color. CHR$(137) desactiva este efecto. Por ejemplo, podemos hacer que las letras azules del ejemplo anterior sean intermitentes mediante:

```
PRINT CHR$(136);CHR$(132);CHR$(157);
      CHR$(135);"EN LA MARINA"
```

La característica más notable de la modalidad 7 es su capacidad para producir caracteres de doble altura. CHR$(141) nos permite hacerlo, pero también debemos imprimir (PRINT) la misma línea dos veces para conseguir el efecto correcto. A los caracteres de doble altura les podemos aplicar todos los otros efectos. Por ejemplo, para producir caracteres azu-

¡Peligro!: campo minado
Estos fotogramas de diversos momentos del juego muestran las minas, el ayudante, el disparo del francotirador, una explosión y las visualizaciones del marcador y los títulos



les intermitentes de doble altura sobre un fondo blanco fijo, usamos las siguientes líneas de código:

```
10 MODE 7
20 PRINT CHR$(141);CHR$(136);CHR$(132);
   CHR$(157);CHR$(135);"EN LA MARINA"
30 PRINT CHR$(141);CHR$(136);CHR$(132);
   CHR$(157);CHR$(135);"EN LA MARINA"
```

Sin embargo, todos estos códigos de control ocupan muchísimo espacio de programa y su digitación lleva mucho tiempo. Un método alternativo consiste en encadenar varios códigos entre sí formando una serie única que se pueda utilizar en las sentencias PRINT. Por ejemplo, si necesitáramos realizar muchos PRINT empleando caracteres rojos sobre fondo amarillo, podríamos empezar por crear una serie (rojo$) que se pudiera luego utilizar en cada sentencia PRINT que requiriera este efecto:

```
10 MODE 7
20 rojo$ = CHR$(129) + CHR$(157) + CHR$(131)
30 PRINTrojo$;"JUEGO TERMINADO"
40 PRINTrojo$;"SU MARCADOR"
```

El procedimiento para el "final del juego"

Examinemos ahora más de cerca cómo usamos estos efectos para la pantalla de final del juego:

```
2110 DEF PROCfinal_juego
2120 IF marcador$ > max_marcador$ THEN max_marcador$ = marcador$
2130 rojo$ = CHR$(129) + CHR$(157) + CHR$(131)
2140 juego$ = "FIN JUEGO"
2150 PRINTTAB(0,5)rojo$;CHR$(141);CHR$(136);TAB(12);juego$
2160 PRINTrojo$;CHR$(141);CHR$(136);TAB(12);juego$
2170 PRINT:PRINTrojo$;"Su marcador";TAB(30);marcador$
2180 PRINT:PRINTrojo$;"Max marcador";TAB(30);max_marcador$
2190 PRINT:PRINTrojo$;"Tiempo";TAB(30);tiempo$
2200 azul$ = CHR$(132) + CHR$(157) + CHR$(134)
2210 go$="OTRA PARTIDA S/N?"
2220 PRINT:PRINT
2230 PRINTazul$;CHR$(141);CHR$(136);TAB(5);go$
2240 PRINTazul$;CHR$(141);CHR$(136);TAB(5);go$
2250 REM ** RESPUESTA ? **
2260 *FX 15,1
2270 respuesta$ = INKEY$(0)
2280 IF GET$ = "N" THEN flag_final = 1
2290 ENDPROC
```

La línea 2120 comprueba si el marcador del juego que acaba de concluir es mayor que el máximo marcador anterior y, caso de ser necesario, actualiza el marcador de puntuación máxima.

El mensaje FIN JUEGO se imprime luego en caracteres intermitentes rojos de doble altura sobre fondo amarillo (líneas 2130 a 2160), y se visualizan los detalles de los marcadores y el tiempo (líneas 2170 a 2190). Se le pregunta entonces al jugador si desea jugar otra vez. Si la respuesta es N, entonces se pone una variable (flag-final) en uno.

Observe que durante este procedimiento no se ha establecido la modalidad 7. Ello se debe a que el BBC Micro no permite efectuar un cambio de modalidad dentro de un procedimiento. Si se intentara, se produciría un mensaje de error BAD MODE (modalidad incorrecta). Debemos, en cambio, establecer la modalidad 7 en el corto programa principal que llama a los procedimientos. Para completar el programa se deben agregar las siguientes líneas. Ahora todo el programa de llamada completo se ha colocado en un bucle REPEAT...UNTIL, que se repetirá hasta que flag-final se ponga a uno.

```
1100 REPEAT

1200 MODE7
1210 REM ** APAGAR CURSOR **
1220 VDU23,1,0;0;0;0;
1230 PROCfinal_juego
1240 UNTIL flag-final = 1
1250 CLS
1260 END
```

La alternativa de Electron

Los usuarios del Electron deben haber ido leyendo nuestro análisis de la modalidad 7 con cierta preocupación, porque el Electron no dispone de la misma. Como alternativa, hemos preparado un procedimiento diferente que utiliza la modalidad 5 para la pantalla del final del juego. Omita la línea 1200 del programa de llamada que acabamos de proporcionar e introduzca este procedimiento en lugar del de final del juego para el BBC:

```
> L.2100,2300
 2100 DEF PROCfinal_juego
 2110 IF marcador$ > max_marcador$ THEN max_marcador$ = marcador$
 2120 REM ASEGURAR FONDO AMARILLO
 2130 VDU19,130,3,0,0,0
 2140 GCOL0,130:REM COLOREAR PANTALLA
 2150 COLOUR1:COLOUR130:REM ESTABLECER COLORES TEXTO
 2160 juego$ = "FIN JUEGO"
 2170 PRINTTAB(2,4);juego$
 2180 COLOUR0
 2190 PRINTTAB(0,8);"Su marcador";TAB(15);marcador$
 2200 PRINT:PRINT"Max marcador";TAB(15);max_marcador$
 2210 PRINT:PRINT"Tiempo";TAB(15);tiempo$
 2220 go$ = "OTRA PARTIDA S/N?"
 2230 REM CAMBIAR COL3 A AMARILLO/AZUL INTERMITENTE
 2240 VDU19,3,11,0,0,0
 2250 COLOUR3
 2260 PRINT:PRINT
 2265 PRINTTAB(2)go$
 2270 REM ** RESPUESTA  ? **
 2275 *FX 15,1
 2280 respuesta$ = INKEY$(0)
 2285 IF GET$ = "N" THEN flag_final = 1
 2290 VDU 20:REM RESTAURAR COLORES POR DEFECTO
 2300 ENDPROC
```

El listado final

```
1000 REM ************************
1010 REM **                    **
1020 REM **       MINAS        **
1030 REM **                    **
1040 REM ************************
1050 :
1060 max_marcador$ = "00000"
1070 flag_final = 0
1080 :
1090 REM **** PROGRAMA PRINCIPAL ****
1100 REPEAT
1110 MODE5
1120 REM ** APAGAR CURSOR **
1130 VDU23;8202;0;0;0;
1140 PROCpagina_titulos
1150 CLS
1160 PROCpreparacion
1170 :
1180 PROCbucle
1190 :
1200 MODE7
1210 REM ** APAGAR CURSOR **
1220 VDU23,1,0;0;0;0;
1230 PROCfinal_juego
1240 UNTIL flag_final = 1
1250 CLS
1260 END
1270 :
1280 :
1290 REM **** DEFINICION DE PROCEDIMIENTOS ****
1300 DEF PROCpagina_titulos
1310 GCOL 0,129
1320 CLG
1330 GCOL 3,3
1340 PROCmusica
1350 Y = 100:X = 0
1360 REPEAT
1370 X = X + 20:Y = Y + 50
1380 FOR I = 1 TO 2
1390 PROCminas
1400 NEXT I
1410 UNTIL Y > 700
1420 :
1430 PROCminas
1440 PRINTTAB(0,20)"Factor de destreza (0-9)?"
1450 PROCmusica
1460 REPEAT
1470 destreza = GET-48
1480 UNTIL destreza > -1 AND destreza < 10
1490 ENDPROC
1500 :
1510 DEF PROCminas
1520 PLOT4,X,Y
1530 REM ** LETRA M **
1540 PLOT1,0,200
1550 PLOT1,80,-100
1560 PLOT1,80,100
1570 PLOT1,0,-200
1580 REM ** LETRA I **
1590 PLOT0,40,0
1600 PLOT1,80,0
1610 PLOT0,-40,0
1620 PLOT1,0,200
1630 PLOT0,-40,0
1640 PLOT1,80,0
1650 REM ** LETRA N **
1660 PLOT0,40,-200
1670 PLOT1,0,200
1680 PLOT1,120,-200
1690 PLOT1,0,200
1700 REM ** LETRA E **
1710 PLOT0,160,0
1720 PLOT1,-120.0
```

Listado final

Este listado muestra claramente la ortografía de los nombres de variables y procedimientos (como max_marcador$ y PROCfinal_juego), que incluyen el carácter de subrayado, "_". Los programadores del BBC Micro ya estarán familiarizados con su utilización como carácter legal de espaciación; no se lo debe confundir con el guión

```
1730 PLOT1,0,-200
1740 PLOT1,120,0
1750 PLOT0,-40,100
1760 PLOT1,-80,0
1770 REM ** LETRA S **
1780 PLOT0,280,60
1790 PLOT 1,0,40
1800 PLOT1,-120,0
1810 PLOT1,0,-100
1820 PLOT1,120,0
1830 PLOT1,0,-100
1840 PLOT1,-120,0
1850 PLOT1,0,40
1860 ENDPROC
1870 :
1880 DEF PROCpreparacion
1890 COLOUR2
1900 flag_final = 0
1910 PROCinicializar_variables
1920 PROCdefinir_caracteres
1930 factor = destreza*3 + 30
1940 PROCcolocar-minas(factor)
1950 PROCtrazar_borde
1960 PROCestablecer_tiempo
1970 PROCestablecer_marcador
1980 PROCestablecer_hombres
1990 PROCsituar_sujetos
2000 ENDPROC
2010 :
2020 DEF PROCbucle
2030 REPEAT
2040 PROCactualizar_tiempo
2050 PROCleer_teclado
2060 rand = RND(50-destreza)
2070 IF rand = 1 THEN PROCfrancotirador
2080 UNTIL TIME > 12099 OR flag_final = 1
2090 ENDPROC
2100 :
2110 DEF PROCfinal_juego
2120 IF marcador$ > max_marcador$ THEN max_marcador$ = marcador$
2130 rojo$ = CHR$(129) + CHR$(157) + CHR$(131)
2140 juego$ = "FIN JUEGO"
2150 PRINTTAB(0,5)rojo$;CHR$(141);CHR$(136);TAB(12);juego$
2160 PRINTrojo$;CHR$(141);CHR$(136);TAB(12);juego$
2170 PRINT:PRINTrojo$;"Su marcador";TAB(30);macador$
2180 PRINT:PRINTrojo$;"Max marcador";TAB(30);max_marcador$
2190 PRINT:PRINTrojo$;"Tiempo";TAB(30);tiempo$
2200 azul$ = CHR$(132) + CHR$(157) + CHR$(134)
2210 go$ = "OTRA PARTIDA S/N?"
2220 PRINT:PRINT
2230 PRINTazul$;CHR$(141);CHR$(136);TAB(5);go$
2240 PRINTazul$;CHRS(141);CHR$(136);TAB(5);go$
2250 REM ** RESPUESTA ? **
2260 *FX 15,1
2270 respuesta$ = INKEY$(0)
2280 IF GET$ = "N" THEN flag_final = 1
2290 ENDPROC
2300 :
2310 REM **** PROCEDIMIENTOS NIVEL 2 ****
2320 DEF PROCinicializar_variables
2330 xdet = 2:ydet = 25:xhom = 17:yhom = 1
2340 xcomienzo = 120:xfinal = 1144
2350 cero$ = "000000"
2360 ENDPROC
2370 :
2380 DEF PROCdefinir_caracteres
2390 REM ** MINAS **
2400 VDU23,224,0,0,56,254,254,124,0,0
2410 REM ** DETECTOR DE MINAS **
2420 VDU23,225,231,195,189,36,36,189,195,231
2430 REM ** AYUDANTE **
2440 VDU23,226,56,56,16,124,186,170,40,108
2450 ENDPROC
2460 :
2470 DEF PROCtrazar_borde
2480 GCOL 0,1
2490 MOVE 120,188
2500 DRAW 120,992
2510 DRAW 1152,992
2520 DRAW 1152,188
2530 DRAW 120,188
2540 ENDPROC
2550 :
2560 DEF PROCcolocar_minas(numero_minas)
2570 REM ** CAMBIAR COLOUR 2 A VERDE **
2580 VDU19,2,2,0,0,0
2590 FOR I = 1 TO numero_minas
2600 PRINTTAB(RND(16) + 1,RND(25));CHR$(224)
2610 NEXT I
2620 ENDPROC
2630 :
2640 DEF PROCestablecer_tiempo
2650 PRINTTAB(2,27)"Tiempo      02:00"
2660 TIME = 0
2670 ENDPROC
2680 :
2690 DEF PROCestablecer_hombres
2700 hombres$ = CHR$(226) + CHR$(226) + CHR$(226)
2710 contador = 1
2720 COLOUR 1
2730 PRINTTAB(2,30);hombres$
2740 COLOUR 2
2750 ENDPROC
2760 :
2770 DEF PROCestablecer  marcador
2780 marcador = 0:marcador$ = "00000"
2790 PRINTTAB(2,28)"Marcador      00000"
2800 PRINTTAB(2,29)"Max marcador";max_marcador$
2810 ENDPROC
2820 :
2830 DEF PROCsituar_sujetos
2840 COLOUR 1
2850 PRINTTAB(xdet,ydet);CHR$(225)
2860 PRINTTAB(xhom,yhom);CHR$(226)
2870 COLOUR 2
2880 ENDPROC
2890 :
2900 DEF PROCactualizar_tiempo
2910 seg$ = STR$(((12100-TIME) DIV 100)MOD 60)
2920 min$ = STR$(((12100-TIME)DIV 6000)MOD 60)
2930 REM ** AGREGAR CEROS POR DELANTE **
2940 seg$ = LEFT$(cero$,2-LEN(seg$)) + seg$
2950 min$ = LEFT$(cero$,2-LEN(min$)) + min$
2960 tiempo$ = min$+":"+ seg$
2970 PRINTTAB(11,27);tiempo$
2980 ENDPROC
2990 :
3000 DEF PROCleer_teclado
3010 REM ** ARRIBA  ? **
3020 IF INKEY(-58) = -1 THEN PROCmover(0,-1)
3030 REM ** ABAJO  ? **
3040 IF INKEY(-42) = -1 THEN PROCmover(0,1)
3050 REM ** DERECHA  ? **
3060 IF INKEY(-122) = -1 THEN PROCmover(1,0)
3070 REM ** IZQUIERDA  ? **
3080 IF INKEY (-26) = -1 THEN PROCmover(-1,0)
3090 ENDPROC
3100 :
3110 DEF PROCfrancotirador
3120 ycomienzo = RND(750) + 220
3130 yfinal = RND(750)+220
3140 dx = 32:dy = (yfinal-ycomienzo)/32
3150 GCOL 3,3
3160 PROClinea
3170 IF POINT(x,y) = 1 THEN PROCexplotar(x,y) ELSE PROClinea
3180 ENDPROC
3190 :
3200 REM **** PROCEDIMIENTOS NIVEL 3 ****
3210 :
3220 DEF PROCmover(delta_x,delta_y)
3230 REM ** BORRAR POSICIONES ANTIGUAS **
3240 COLOUR 1
3250 PRINTTAB(xdet,ydet);" "
3260 PRINTTAB(xhom,yhom);" "
3270 REM ** MOVER DETECTOR **
3280 xdet = xdet + delta_x
3290 ydet = ydet + delta_y
3300 REM ** COMPROBAR LIMITES **
3310 IF xdet > 17 THEN xdet = 17
3320 IF ydet > 25 THEN ydet = 25
3330 IF xdet < 2 THEN xdet = 2
3340 IF ydet < 1 THEN ydet = 1
3350 REM ** CALCULAR COORDENADAS DEL HOMBRE **
3360 xhom = 19-xdet
3370 yhom = 26-ydet
3380 PROCconvertir(xhom,yhom)
3390 IF POINT(xgraf,ygraf) = 2 THEN PROCexplotar(xgraf,ygraf)
3400 PROCconvertir(xdet,ydet)
3410 IF POINT(xgraf,ygraf) = 2 THEN PROCdescubierta_mina
3420 PROCsituar_sujetos
3430 ENDPROC
3440 :
3450 DEF PROClinea
3460 SOUND0,-8,4,5
3470 x = xcomienzo:y = ycomienzo
3480 MOVE x,y
3490 REPEAT
3500 DRAW x,y
3510 x = x + dx:y = y + dy
3520 UNTIL x > xfinal OR POINT(x,y) = 1
3530 ENDPROC
3540 :
3550 DEF PROCexplotar(x_explosion,y_explosion)
3560 REM ** EFECTO SONORO **
3570 SOUND 0,-15,6,50
3580 REM ** ESTABLECER VELOCIDAD FLASH **
3590 *FX9,20
3600 *FX10,50
3610 FOR I = 1 TO 100
3620 MOVE x_explosion,y_explosion
3630 VDU19,2,RND(15),0,0,0
3640 GCOL 0,RND(3)
3650 PLOT 1,RND(100)-50,RND(100)-50
3660 NEXT I
3670 PROCrestaurar
3680 ENDPROC
3690 :
3700 REM ** PROCEDIMIENTOS NIVEL 4 ****
3710 :
3720 DEF PROCconvertir(xcar,ycar)
3730 xgraf = 64*xcar + 32
3740 ygraf = 1023-(32*ycar + 16)
3750 ENDPROC
3760 :
3770 DEF PROCdescubierta_mina
3780 REM ** EFECTO SONORO **
3790 SOUND 2,-15,170,3
3800 REM ** INCREMENTAR MARCADOR **
3810 COLOUR 2
3820 marcador = marcador + 150
3830 marcador$ = STR$(marcador)
3840 marcador$ = LEFT$(cero$,5-LEN(marcador$)) + marcador$
3850 PRINTTAB(11,28);marcador$
3860 ENDPROC
3870 :
3880 DEF PROCrestaurar
3890 contador = contador + 1
3900 IF contador > 4 THEN flag_final = 1: ENDPROC
3910 CLS
3920 VDU19,2,2,0,0,0
3930 COLOUR 2
3940 PROCinicializar_variables
3950 minas_restantes = factor-marcador/150
3960 PROCcolocar_minas(minas_restantes)
3970 PROCtrazar_borde
3980 PRINTTAB(2,27);"Tiempo"
3990 PRINTTAB(2,28)"Marcador"
4000 PRINTTAB(11,28)marcador$
4010 PRINTTAB(2,29)"Max marcador"
4020 PRINTTAB(11,29)max_marcador$
4030 vidas_restantes$ = LEFT$(hombres$,4-contador)
4040 COLOUR 1
4050 PRINTTAB(2,30);vidas_restantes$;"   "
4060 COLOUR 2
4070 PROCsituar_sujetos
4080 ENDPROC
4090 DEF PROCmusica
4100 REM ** PRIMERA BARRA **
4110 SOUND 1,-8,213,5
4120 SOUND 1,-8,209,5
4130 SOUND 1,-8,213,5
4140 SOUND 1-8,209,5
4150 SOUND 1,-8,213,5
4160 SOUND 1-8,193,5
4170 SOUND 1,-8,205,5
4180 SOUND 1,-8,197,5
4190 REM ** SEGUNDA BARRA **
4200 SOUND1,-8,185,20
4210 SOUND1,-8,165,5
4220 SOUND1,-8,185,5
4230 SOUND1,-8,193,20
4240 REM ** TERCERA BARRA **
4250 SOUND1,-8,165,5
4260 SOUND1,-8,193,5
4270 SOUND1,-8,197,20
4280 ENDPROC
```

Jinetes de la noche

La inteligente combinación de elementos de juegos de guerra y de aventuras con sus avanzadas técnicas de gráficos hacen que este juego marque pautas para el futuro

Los juegos de guerra por ordenador, recreaciones de juegos de tablero como el *Blitzkrieg* y el *Diplomacy*, difieren considerablemente de los juegos de aventuras, la mayoría de los cuales se basan, aunque de forma bastante libre, en el clásico *Dungeons and dragons* (Calabozos y dragones). Un juego de guerra exige la aplicación de estrategia y planes tácticos, con todas las piezas (ejércitos, suministros, armas, etc.) visualizadas en un mapa del campo de batalla. Un juego de aventuras se basa en la sorpresa y el ingenio: un jugador debe resolver una serie de problemas que se le presentan de uno en uno a medida que el juego va avanzando en el escenario. En *Lords of Midnight* (Caballeros de medianoche), para el Spectrum de 48 Kbytes, Beyond Software ha combinado las dos formas para producir una nueva clase de juego.

Al jugador se le proporciona un folleto de 30 páginas que contiene un mapa de la Tierra de Medianoche, en la cual se desarrolla la acción. El jugador representa a Luxor the Moonprince, Lord of the Free (Luxor, príncipe de la Luna y señor de la Libertad). Luxor está en posesión del "anillo de la luna", dispositivo que le permite controlar (y ver a través de los ojos de ellos) a sus cuatro compañeros y a cualquier señor de la Libertad que pueda reclutar. Los señores que la Libertad tienen bajo su control la mayor parte del sur, que se debe defender contra el ataque de Doomdark, the Witchking of Midnight (el rey brujo de medianoche). Las fuerzas de Doomdark las controla el ordenador; con base en una ciudadela situada en el norte, ellos intentan controlar las regiones sureñas.

Doomdark dispone de la ayuda de la "corona de hielo", dispositivo mágico que arroja sobre sus enemigos el "temor glacial". Éste hace disminuir las fuerzas de cualquier señor de la Libertad que se aproxime a las tierras del norte. Sin embargo, uno de sus compañeros, Morkin, es inmune al "temor glacial", de modo que su misión consiste en avanzar sin desmayo hacia el norte para destruir la "corona", mientras los otros personajes se empeñan en luchar abiertamente contra las fuerzas de Doomdark.

Los juegos de guerra tradicionales se basan en un mapa que permite a los jugadores llevar el registro de las fuerzas desplegadas contra ellos. En *Lords of midnight*, los jugadores no disponen de este mapa constantemente actualizado, sino que éste está retenido en la memoria del Spectrum. La panorámica de la acción de que dispone el jugador es la que tiene ante sus ojos un señor de la Libertad, que puede mirar en cualquiera de ocho direcciones que se seleccionan desde el teclado. En consecuencia, si hubiera un ejército enemigo aguardando detrás de la serie de colinas que tiene delante, sería incapaz de verlo a menos que las rodease y mirara en la dirección correcta. Esto confiere a los juegos de guerra una nueva dimensión.

La característica más destacada de *Lords of midnight* son los notables gráficos del juego. Beyond Software afirma que hay 32 000 escenas diferentes y 32 personajes con los cuales se las puede ver. Obviamente, el Spectrum no puede retener tal cantidad de pantallas en su memoria, de modo que Beyond ha desarrollado una técnica llamada "paisajística". Los miles de diversas visualizaciones en pantalla se componen todas de 15 formas diferentes, cada una de las cuales está disponible en cuatro tamaños distintos para dar idea de perspectiva. Estas formas básicas se combinan para posibilitar la construcción de visualizaciones complejas: colinas, bosques, montañas, ejércitos, pueblos y ciudadelas, todo ello ilustrado de forma detallada. La memoria del Spectrum retiene un mapa que proporciona la posición de los elementos en cualquiera de 4 000 ubicaciones. Esta información permite que el ordenador calcule la panorámica desde cualquier punto dado.

Lords of midnight responde a una concepción y una presentación hermosísimas. Para mantenerse a tono con el tema del juego, Beyond incluso ha rediseñado el juego de caracteres del Spectrum para darle cierto sabor gótico a los mensajes del texto.

Lords of midnight: Para el Spectrum de 48 K
Editado por: Beyond Software, Competition House, Farndon Road, Market Harborough, Leicestershire LE19 9NR, Gran Bretaña
Autor: Mike Singleton
Palancas de mando: No se necesitan
Formato: Cassette





Efecto gótico
Según Beyond Software, en el juego hay 32 000 escenas diferentes, ¡aunque dudamos que alguien haya intentado contarlas alguna vez! Observe las letras góticas del texto. Beyond ha rediseñado el juego de caracteres del Spectrum para darle un aire más "teutónico"

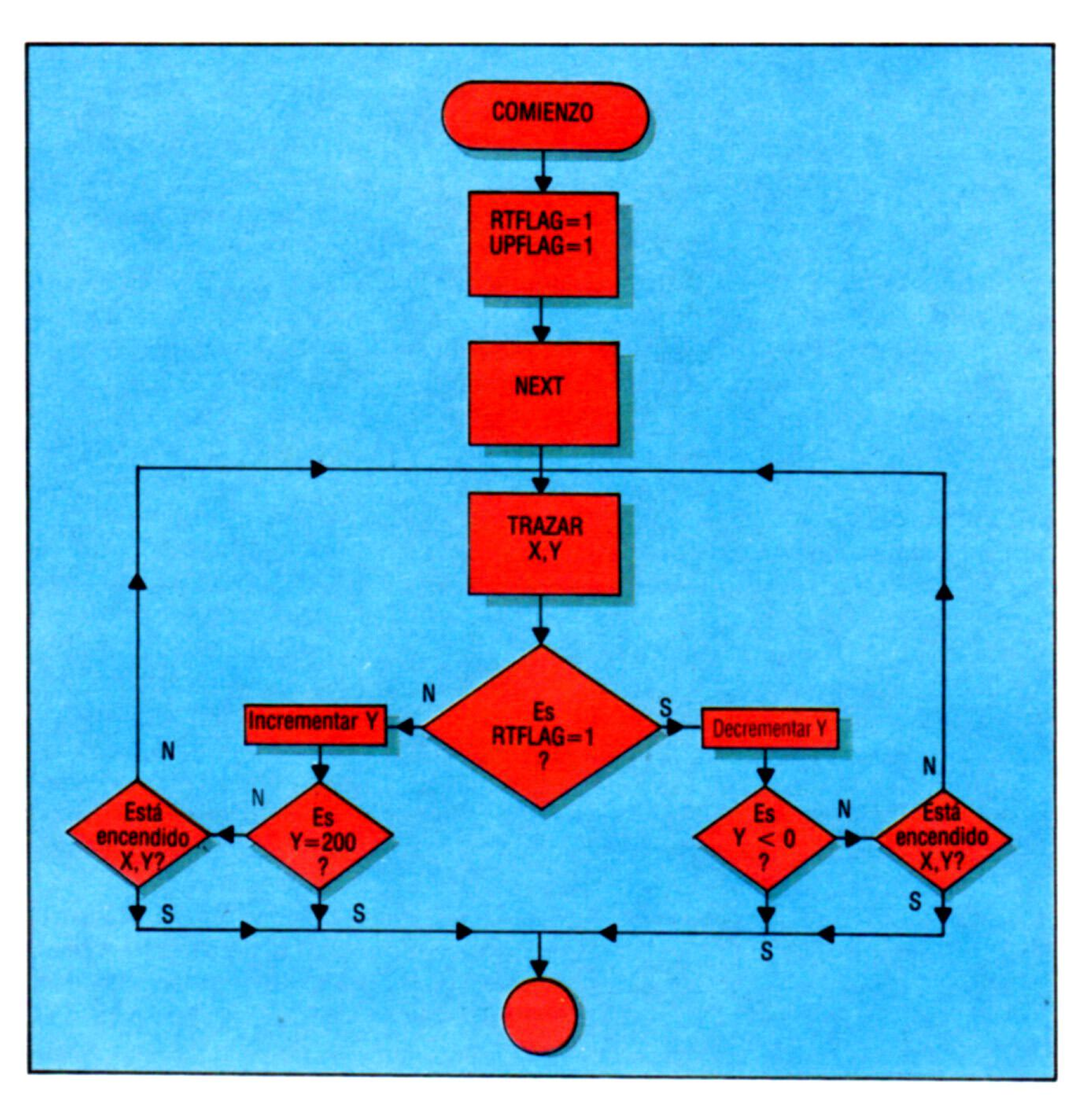


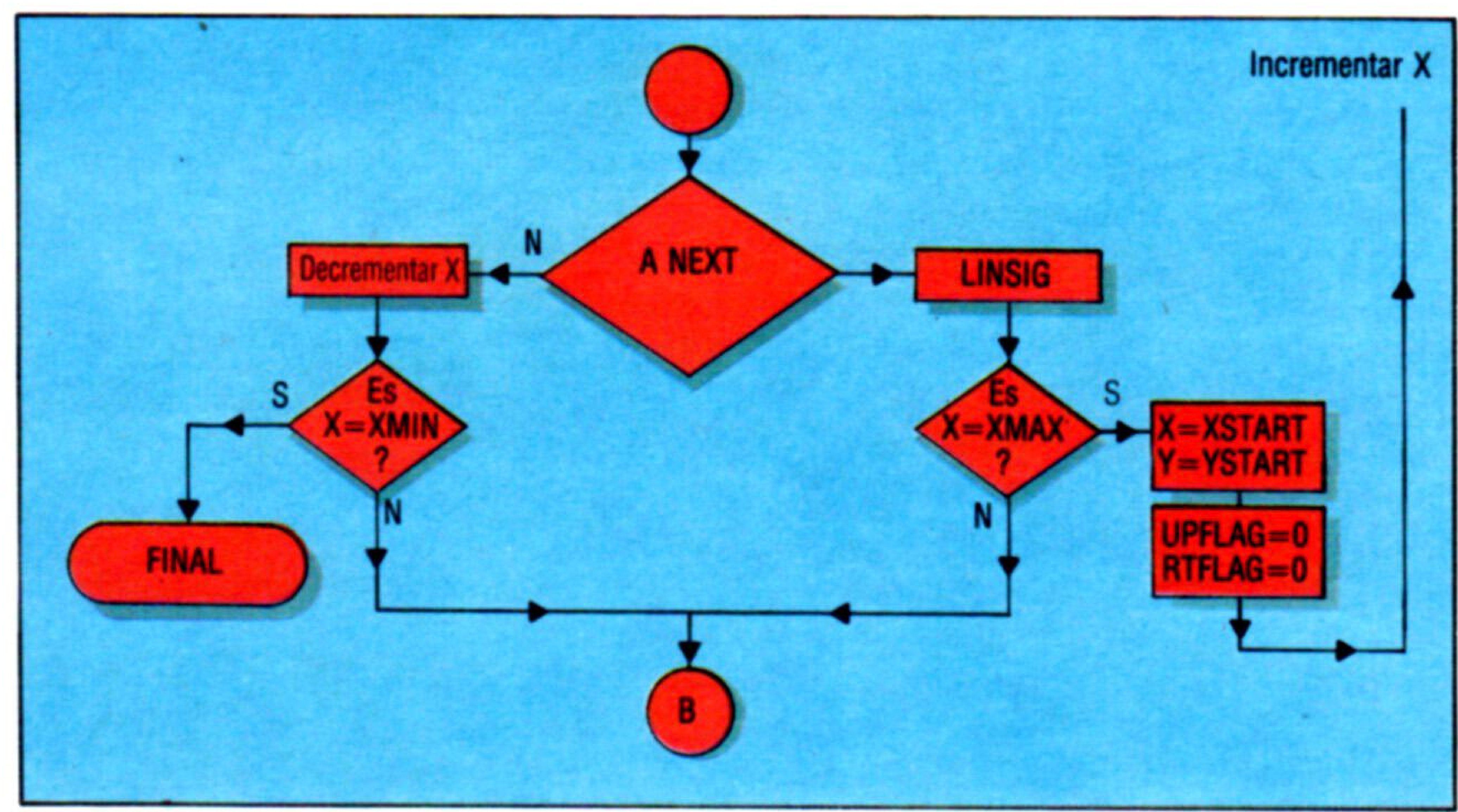
Liz Heaney

Ficha indicadora
Lords of midnight viene con una ficha para colocar encima del teclado, mediante la cual se le simplifica al jugador la tarea de buscar las teclas correctas para las instrucciones

Los puntos crecen

Disponemos ya de la rutina del punto, de la línea y del círculo. Nos falta la rutina del relleno (Fillsub), que vamos a analizar en este capítulo y que resulta bastante más compleja de lo que uno pudiera imaginar





Hay muchos modos de estructurar un algoritmo para rellenar siluetas en la pantalla, pero aunque esto al principio parece tarea fácil resulta más complicado de lo que uno pueda imaginarse. Las figuras que incluyen ángulos "reflejos" (ángulos interiores mayores de 180°) o las que se contraen y se expanden presentan sus dificultades propias. Es posible preparar una rutina que resuelva algunas de estas dificultades, pero no todas. No es cosa fácil dar a un programa la "inteligencia" suficiente para captar lo que constituye una figura cerrada.

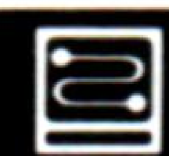
El método que usaremos para nuestra rutina comienza rellenando la figura desde un punto inicial, determinado por el usuario y situado en cualquier lugar dentro del contorno de la figura. La rutina empieza entonces por avanzar pantalla arriba, trazando puntos hasta alcanzar el borde de la figura, a partir del cual se desplaza hacia la derecha un pixel nada más para descender pantalla abajo hasta topar con un nuevo borde. Vuelve a desplazarse entonces un pixel a la derecha para tornar a subir. Este proceso acaba cuando la parte de la figura está rellenada. Pero se vuelve a iniciar en el punto de comienzo moviéndose ahora hacia la izquierda y siempre de arriba abajo y viceversa.

La primera parte de la rutina es inmediata. Dos flags, UPFLAG y RTFLAG (*up*: arriba; *right*: derecha), sirven para indicar la dirección del movimiento de la rutina en cada momento del programa. El primer fragmento del diagrama de flujo muestra la parte de la rutina que controla los incrementos de puntos. El bucle principal incrementa o decrementa el valor de la ordenada Y según el estado de UPFLAG.

Después de preguntar si se trata del borde de la pantalla y comprobar si el pixel siguiente ya está encendido, la rutina cierra el bucle y traza el punto siguiente. Si se encuentra con un pixel ya encendido y con el borde de la pantalla, entonces pasa a la fase siguiente.

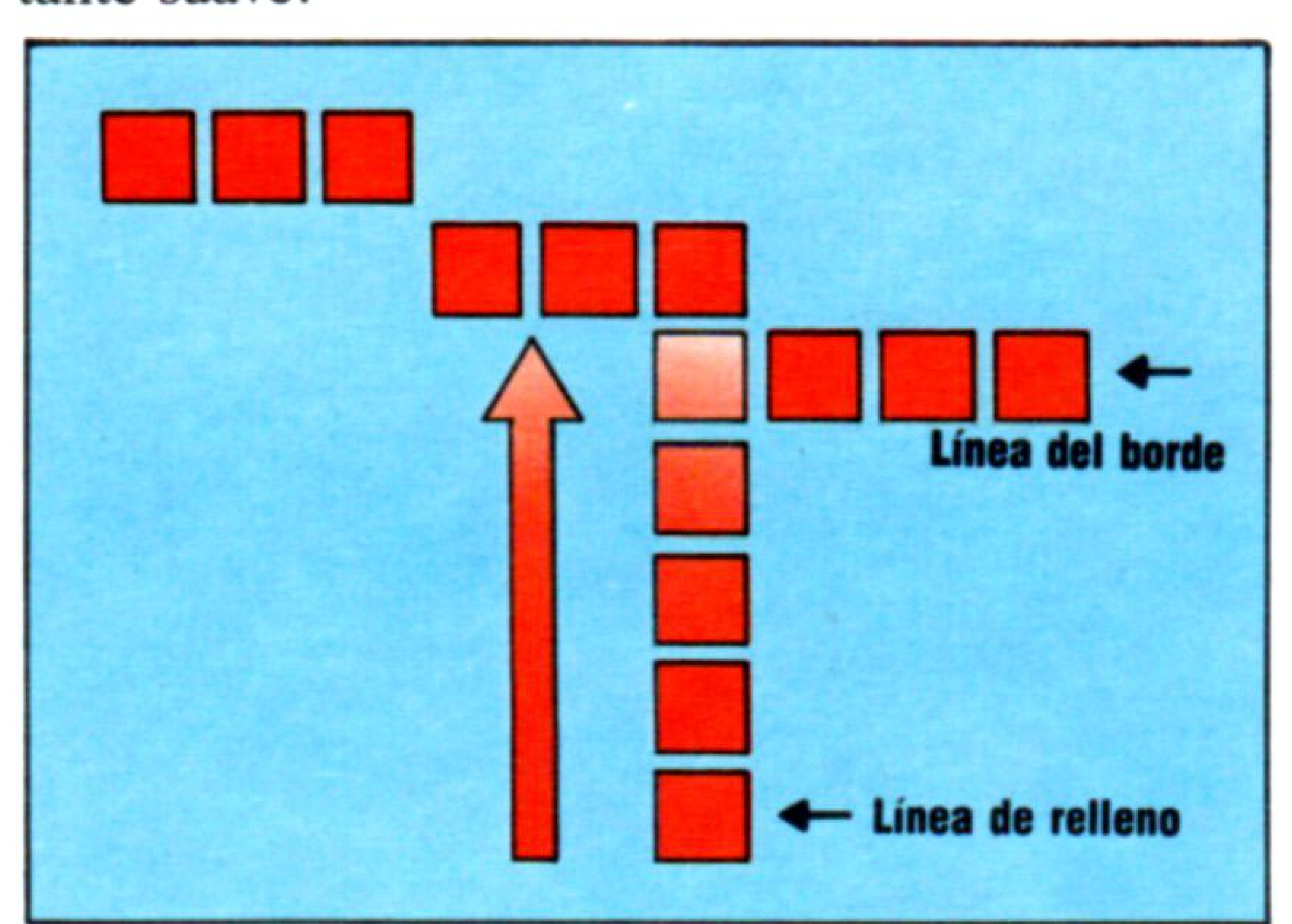
La rutina va ahora a moverse a la derecha o a la izquierda según se lo indique el estado de RTFLAG. Es difícil detectar el borde izquierdo o derecho de la figura. La rutina, en vez de hacerlo, deja que el usuario establezca por sí mismo valores máximo y mínimo para la abscisa X. Esto ofrece además la posibilidad de rellenar franjas dentro de la figura.

El segundo diagrama de flujo nos muestra cómo los valores de X e Y son restablecidos con los valores iniciales en caso de que, moviéndose a la derecha, alcance el valor máximo de X. Entonces los flags de dirección se ponen a cero como preparativo para rellenar la parte izquierda de la figura. Si el valor de X disminuye hasta alcanzar XMIN, quiere decir que la rutina está acabada. Pero si no se han alcanzado todavía los valores máximos y mínimos de X, la rutina continuará para rellenar la línea siguiente.

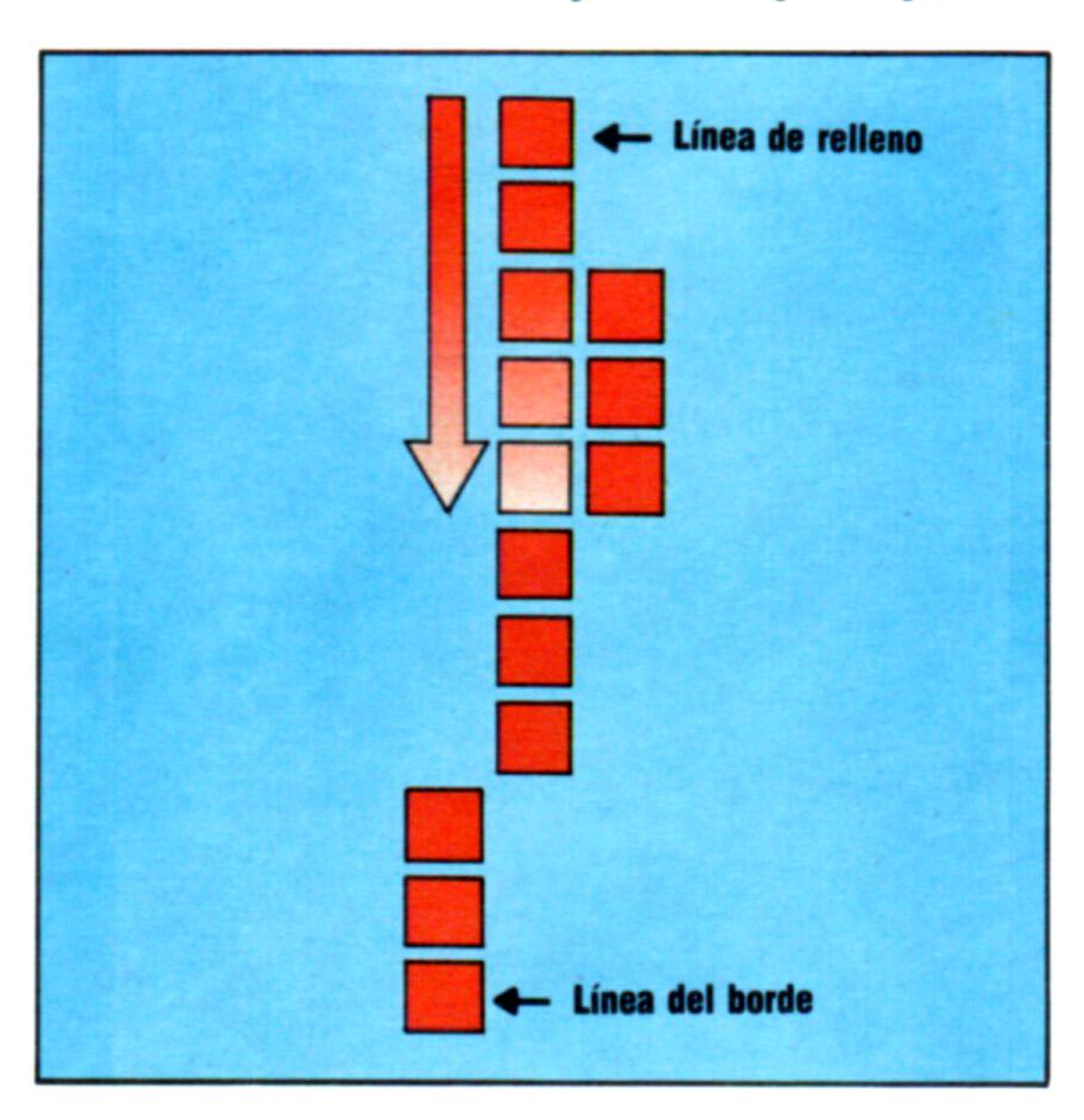


Problemas de pendientes

El algoritmo que abarque todo tipo posible de líneas (muy inclinadas o poco, más o menos gruesas) parece que deba ser complicado, pero los principios operativos son relativamente simples. Pensemos en el caso en que la rutina se está moviendo pantalla arriba y encuentra una línea con una pendiente bastante suave.



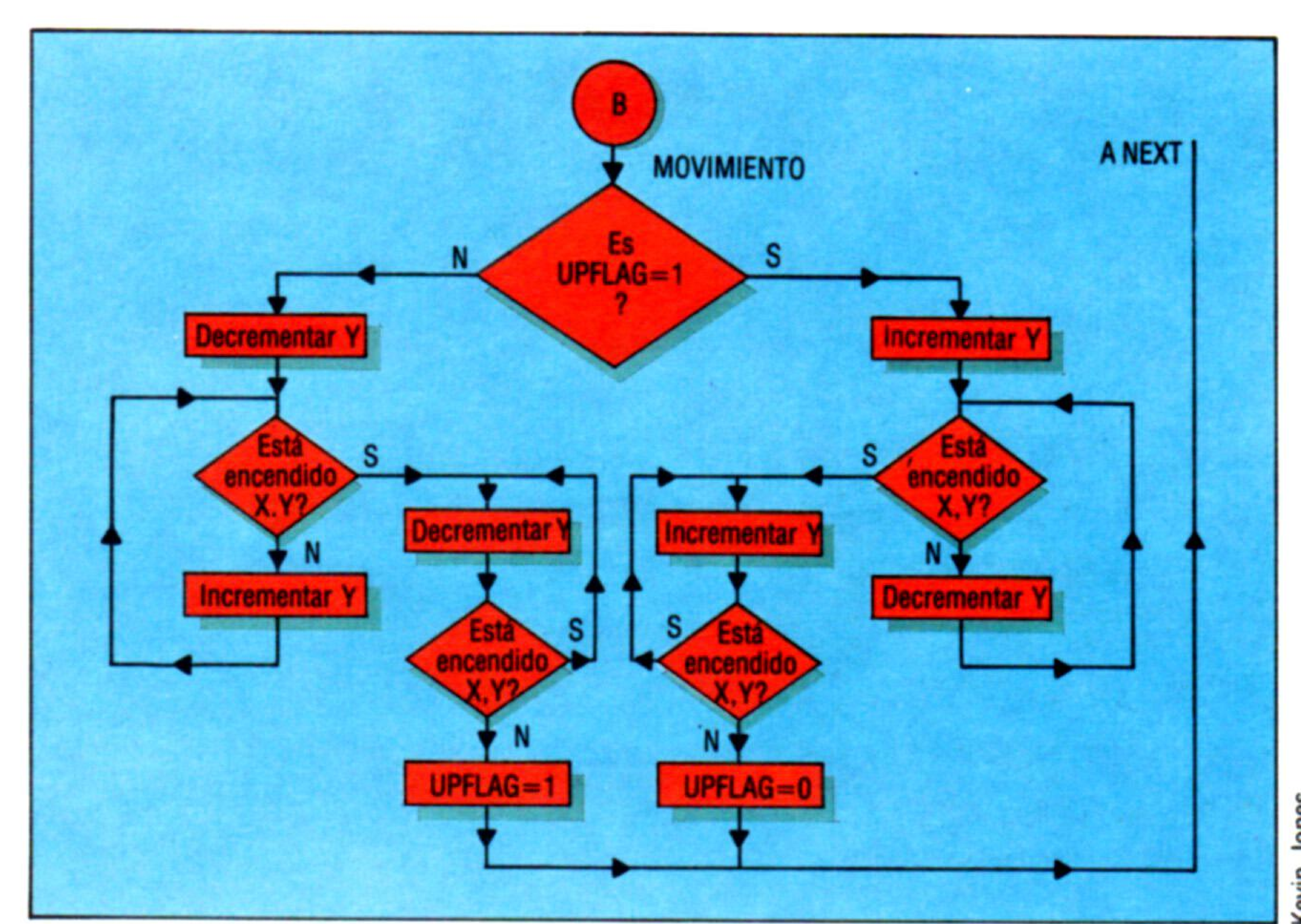
Lo primero que hace el algoritmo al encontrar la línea es retroceder un pixel. Si el movimiento fue a la derecha, éste debe repetirse de modo que pueda iniciarse la siguiente línea en un pixel no iluminado. En el caso de un movimiento descendente de la línea de relleno que encuentra una línea muy inclinada, estaremos ante otro problema por superar.



Si el relleno ha de moverse a la derecha, debe retroceder tres pixels hasta encontrar un espacio vacío desde el cual comenzar su relleno hacia arriba. Pero si ha de moverse a la izquierda descenderá tres pixels más de modo que pueda iniciar su relleno hacia arriba en la primera posición no encendida sobre la línea de borde. Por tanto, para cada dirección de relleno debemos preparar dos bucles para que pueda moverse en una de las dos direcciones hasta la primera posición no encendida. Arriba, a la derecha, diagrama de flujo de este algoritmo.

Listado de la rutina

El cuerpo principal de la rutina Fillsub (*fill*: relleno) sigue bastante de cerca el diagrama de flujo diseñado.



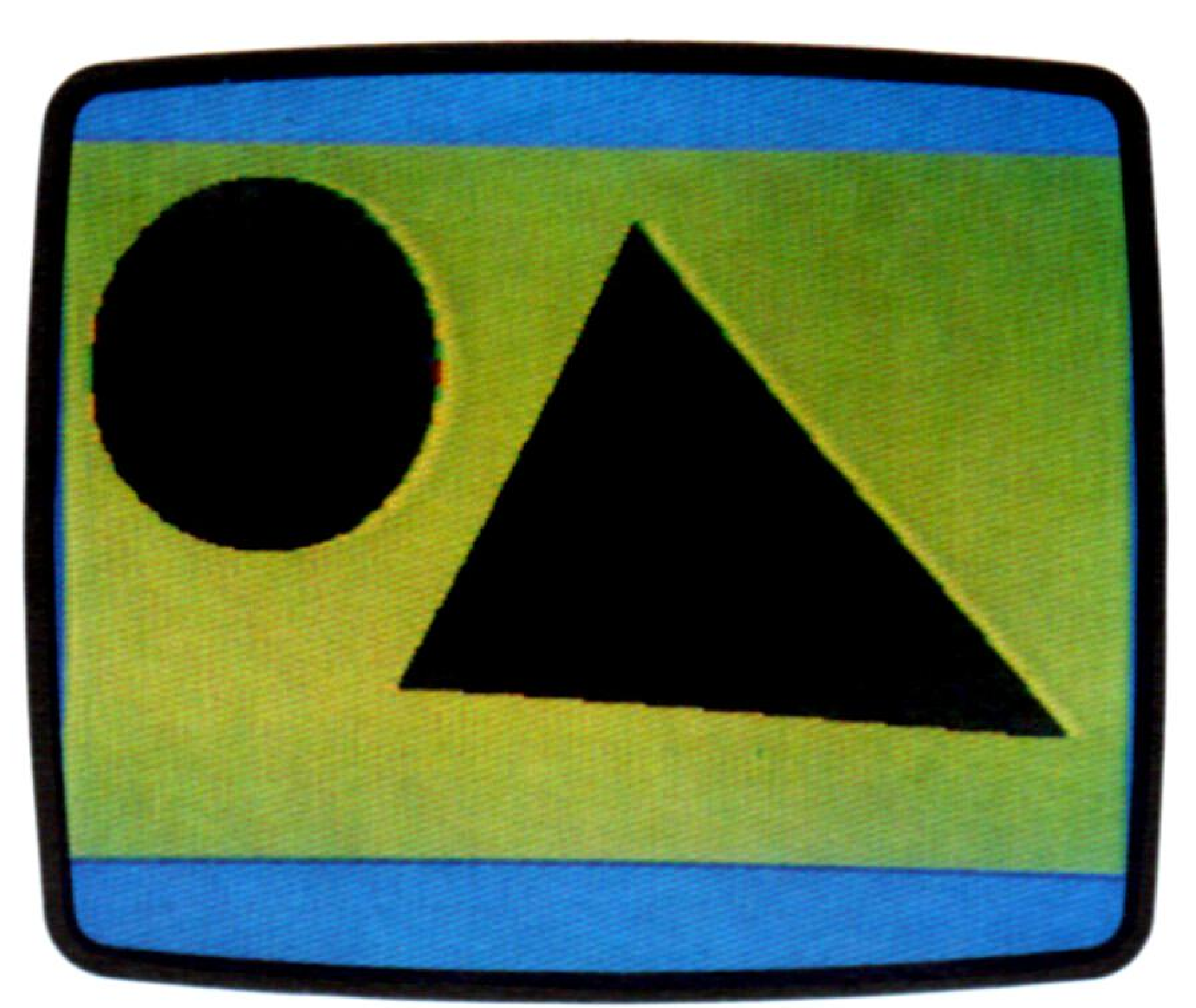
Kevin Jones

Las etiquetas que se usaron en el listado del código fuente se han incluido en las posiciones correspondientes del diagrama para ayudarle a encontrar el recorrido a través del programa.

En varios momentos el algoritmo del relleno necesita comprobar si el punto está encendido o no. El programa hace esto con una rutina llamada POINT (punto), que calcula la dirección del punto desde sus coordenadas X,Y de la misma forma que nuestro programa Plotsub. Pero en lugar de realizar un OR lógico para poner el bit a uno, lo que hace POINT es usar AND para ver si el bit en cuestión es uno o no. Si el resultado de la operación AND no es cero, entonces el bit es uno. Este resultado se almacena en PTFLAG para una verificación posterior de parte del programa principal. POINT en gran parte hace el mismo trabajo que Plotsub y los lectores que ya tienen algún dominio del lenguaje del 6502 puede que deseen modificar Plotsub para que encienda un bit en el mapa de bits además de preguntar por el valor de cada bit.

Veamos, por último, cómo se utiliza la rutina Fillsub. Ante todo, Fillsub necesita que se le suministren varios parámetros. Éstos son:

- Las coordenadas del punto inicial. Este punto *debe* estar dentro de la figura que se desea rellenar.
- Los valores máximos y mínimos de la abscisa X. Teniendo buen cuidado con las figuras con ángulos muy agudos, ya que Fillsub puede moverse fuera



Valioso esfuerzo
Después de estos últimos capítulos y una vez mecanografiados varios kilobytes de lenguaje máquina, llegamos finalmente a conseguir para el Commodore 64 esa facilidad gráfica en alta resolución que cualquier otro microordenador familiar consigue por medio del BASIC. Este esfuerzo le ahorra varios billetes de banco en el caso de que usted no desee utilizar el cartucho de BASIC ampliado

de la figura si llega al borde del espacio interior antes de alcanzar uno de los límites establecidos. El problema se resuelve poniendo los límites un poco más hacia dentro. Observe que la abscisa X del punto inicial y los límites de X deben separarse en la forma byte *hi* / byte *lo*, como se ilustra en el programa de demostración.

Aunque Fillsub no se apoya en ninguna otra rutina ya expuesta para el Commodore 64, las restantes tres rutinas (Plotsub, Linesub y Circsub) también las carga el programa de demostración para dibujar las figuras que ha de rellenar Fillsub.

Cuando esta rutina se creó por primera vez, el relleno se realizaba mediante líneas horizontales y no verticales. Pero se descubrió que el empleo de barras verticales para rellenar figuras se traducía en una ejecución considerablemente más rápida.

Cargador Plotsub/II

Ésta es una versión corregida de la rutina Plotsub que publicamos por primera vez en la página 819. Empléela para crear un nuevo archivo objeto llamado "PLOTSUB.HEX" en cinta o cassette, según se explicó en la citada página

```
10 FORI = 49408TO49408+314
20 READA:POKEI,A;S=S+A: NEXT
30 READCC:IFCC<>STHENPRINT"ERROR SUMA CONTROL"
100 DATA1,0,3,6,0,5,0,0,6,5,5,69,38,2
110 DATA72,138,72,152,72,173,0,193,240
120 DATA83,169,0,133,251,169,4,133,252
130 DATA162,3,160,0,173,2,193,145,251
140 DATA136,208,251,230,252,202,48,8
150 DATA208,244,145,251,160,231,208
160 DATA238,173,1,193,240,24,169,0,133
170 DATA251,169,32,133,252,162,32,160
180 DATA0,169,0,145,251,136,208,251
190 DATA230,252,202,208,246,173,24,208
200 DATA41,240,9,8,141,24,208,173,17
210 DATA208,9,32,141,17,208,76,125,193
220 DATA173,24,208,41,240,9,4,141,24
230 DATA208,173,17,208,41,223,141,17
240 DATA208,104,168,104,170,104,96,72
250 DATA138,72,152,72,173,4,193,141,7
260 DATA193,173,3,193,41,248,141,6,193
270 DATA173,3,193,41,7,141,8,193,173,5
280 DATA193,41,7,141,10,193,162,3,78,5
290 DATA193,202,208,250,173,5,193,141
300 DATA,9,193,169,0,141,11,193,141,12
310 DATA193,162,5,173,11,193,24,109,9
320 DATA193,141,11,193,202,208,243,162
330 DATA6,14,12,193,14,11,193,144,3
340 DATA238,12,193,202,208,242,173,11
350 DATA193,24,109,6,193,141,11,193
360 DATA173,12,193,109,7,193,141,12
370 DATA193,173,11,193,24,105,0,141,11
380 DATA193,173,12,193,105,32,141,12
390 DATA193,173,11,193,24,109,10,193
400 DATA141,11,193,173,12,193,105,0
410 DATA141,12,193,173,11,193,133,251
420 DATA173,12,193,133,252,169,1,141
430 DATA13,193,56,169,7,237,8,193,240
440 DATA7,170,14,13,193,202,208,248
450 DATA160,0,177,251,13,13,193,145
460 DATA251,76,125,193
470 DATA37523:REM*SUMA DE CONTROL*
```

Artificio extraño
La línea 15 DN = 8 indica que los archivos objeto (Plotsub.Hex, etc.) deben cargarse desde un disco. Para usar una cinta cámbiese por DN = 1, y después, haga una cinta con los archivos objeto según el orden especificado en las líneas 20 a 30, o bien, si sus archivos están en cintas diferentes, coloque este código como líneas 22, 26 y 28: INPUT "CAMBIE CINTA Y PULSE RETURN";A$

Demostración Fillsub

```
10 REM *** PROGRAMA DEMO FILLSUB ****
15 DN=8:REM PARA CASSETTE DN=1
20 IFA=0THENA=1:LOAD"PLOTSUB.HEX",DN,1
25 IFA=1THENA=2:LOAD"LINESUB.HEX",DN,1
27 IFA=2THENA=3:LOAD"CIRCSUB.HEX",DN,1
30 IFA=3THENA=4:LOAD"FILLSUB.HEX",DN,1
40 GOSUB1000:REM ESTABLECE ALT RES
50 REM **** DIBUJA TRIANGULO ****
60 XA=100:YA=150:XB=300:YB=160:XC=170:YC=20
80 X1=XA:Y1=YA:X2=XB:Y2=YB:GOSUB2000
90 X1=XC:Y1=YC:GOSUB2000
100 X2=XA:Y2=YA:GOSUB2000
102 REM **** DIBUJA CIRCULO ****
103 XC=60:YC=60:R=50:GOSUB4000
120 REM **** RELLENA TRIANGULO ****
130 XS=170:YS=130:REM COORDENADAS INICIO
140 MIN=100:MAX=299:REM LIMITES
150 GOSUB3000
161 REM **** RELLENA CIRCULO ****
162 XS=60:YS=60:REM COORDENADAS INICIO
163 MIN=10:MAX=109
164 GOSUB3000
```

Demostración Fillsub (cont.)

```
200 GETA$:IFA$=""THEN200:REM ESPERA PULSAR TECLA
210 POKE49408,0:SYS49422:REM RESTABLECE PANTALLA
220 PRINTCHR$(147):REM BORRA PANTALLA
225 PRINT"FIN DE RUTINA"
230 END
1000 REM **** ESTABLECE ALT RES ****
1010 POKE49408,1:POKE49409,1
1020 POKE49410,7
1030 SYS49422
1040 RETURN
2000 REM **** LINESUB ****
2010 MHI=INT(X1/256):MLO=X1-256*MHI
2020 NHI=INT(X2/256):NLO=X2-256*NHI
2030 POKE49920,MLO:POKE49921,MHI
2040 POKE49922,NLO:POKE49923,NHI
2050 POKE49924,Y1:POKE49925,Y2
2060 SYS 49934
2070 RETURN
3000 REM **** FILLSUB ****
3010 SH=INT(XS/256):SL=XS-SH*256
3020 HAX=INT(MAX/256):LAX=MAX-256*HAX
3030 HIN=INT(MIN/256):LIN=MIN-256*HIN
3040 POKE50955,SL:POKE50956,SH
3050 POKE50957,YS
3060 POKE50958,LIN:POKE50959,HIN
3070 POKE50960,LAX:POKE50961,HAX
3080 SYS50967
3090 RETURN
4000 REM **** CIRCSUB ****
4010 CHI=INT(XXC/256):CLO=XC-256*CHI
4020 POKE50497,CLO:POKE50498,CHI
4030 POKE50499,YC:POKE50500,R
4040 SYS 50521
4060 RETURN
```

Cargador de Fillsub

```
10 REM **** CARGADOR EN BASIC DE FILLSUB ****
20 FORI=50944 TO 51375
30 READA:POKEI,A:CC=CC+A:NEXT
40 READA:IFCC<>A THEN PRINT"ERROR SUMA CONTROL":END
100 DATA11,0,6,8,0,3,6,5,136,39,16,60
110 DATA0,60,10,0,109,0,0,1,10,0,16
120 DATA173,11,199,141,20,199,173,12
130 DATA199,141,21,199,172,13,199,169
140 DATA1,141,18,199,141,19,199,140,5
150 DATA193,173,20,199,141,3,193,173
160 DATA21,199,141,4,193,32,131,193
170 DATA173,19,199,208,8,200,192,200
180 DATA240,19,76,82,199,136,192,0,144
190 DATA11,32,246,199,173,22,199,208,3
200 DATA76,46,199,173,18,199,208,31
210 DATA173,20,199,56,233,1,141,20,199
220 DATA173,21,199,233,0,141,21,199
230 DATA205,15,199,208,65,173,20,199
240 DATA205,14,199,208,57,96,173,20
250 DATA199,24,105,1,141,20,199,173,21
260 DATA199,105,0,141,21,199,205,17
270 DATA199,208,34,173,20,199,205,16
280 DATA199,208,26,173,11,199,141,20
290 DATA199,173,12,199,141,21,199,172
300 DATA13,199,169,0,141,19,199,141,18
310 DATA199,76,46,199,173,19,199,208
320 DATA28,136,32,246,199,173,22,199
330 DATA208,4,200,76,191,199,238,19
340 DATA199,136,32,246,199,173,22,199
350 DATA208,247,76,46,199,200,32,246
360 DATA199,173,22,199,208,4,136,76
370 DATA219,199,206,19,199,200,32,246
380 DATA199,173,22,199,208,247,76,46
390 DATA199,72,138,72,152,72,140,2,199
400 DATA173,20,199,141,0,199,173,21
410 DATA199,141,1,199,173,1,199,141,4
420 DATA199,173,0,199,41,248,141,3,199
430 DATA173,0,199,41,7,141,5,199,173,2
440 DATA199,41,7,141,7,199,162,3,78,2
450 DATA199,202,208,250,173,2,199,141
460 DATA6,199,169,0,141,8,199,141,9
470 DATA199,162,5,173,8,199,24,109,6
480 DATA199,141,8,199,202,208,243,162
490 DATA6,14,8,199,46,9,199,202,208
500 DATA247,173,8,199,24,109,3,199,141
510 DATA8,199,173,9,199,109,4,199,141
520 DATA9,199,173,8,199,24,105,0,141,8
530 DATA199,173,9,199,105,32,141,9,199
540 DATA173,8,199,24,109,7,199,133,251
550 DATA173,9,199,105,0,133,252,169,1
560 DATA141,10,199,56,169,7,237,5,199
570 DATA240,7,170,14,10,199,202,208
580 DATA250,160,0,177,251,45,10,199
590 DATA141,22,199,104,168,104,170,104
600 DATA96
610 DATA50785 REM*SUMA CONTROL*
```



Listado assembly

```
                 ;++++++++++++++++
                 ;++++++++++++++++
                 ;++             ++
                 ;++  FILLSUB 64 ++
                 ;++             ++
                 ;++++++++++++++++
                 ;++++++++++++++++
                 ;
                 ;++++ VALORES PLOTSUB ++++
                 ;
                 PLTSUB  =$C183
                 XLO     =$C103
                 XHI     =$C104
                 YLO     =$C105
                 MPBLO   =$00
                 MPBHI   =$20
                 PTR     =$FB
                 *       =$C700
                 ;
                 ;++++ VARIABLES FILLSUB ++++
                 ;
                 PXLO    *=*+1
                 PXHI    *=*+1
                 PYLO    *=*+1
                 PHBLO   *=*+1
                 PHBHI   *=*+1
                 PREMX   *=*+1
                 PVBYTE  *=*+1
                 PREMY   *=*+1
                 PROWLO  *=*+1
                 PROWHI  *=*+1
                 PBPOS   *=*+1
                 ;
                 XSTLO   *=*+1
                 XSTHI   *=*+1
                 YST     *=*+1
                 XMINLO  *=*+1
                 XMINHI  *=*+1
                 XMAXLO  *=*+1
                 XMAXHI  *=*+1
                 RTFLAG  *=*+1
                 UPFLAG  *=*+1
                 FXLO    *=*+1
                 FXHI    *=*+1
                 PTFLAG  *=*+1
                 ;
                 ;++++ PUNTOS DE INICIO A FX,FY ++++
                 ;
AD 0B C7                 LDA  XSTLO
8D 14 C7                 STA  FXLO
AD 0C C7                 LDA  XSTHI
8D 15 C7                 STA  FXHI
AC 0D C7                 LDY  YST
                 ;
                 ;++++ ESTABLECE FLAGS ++++
                 ;
A9 01                    LDA  #$01
8D 12 C7                 STA  RTFLAG
8D 13 C7                 STA  UPFLAG
                 ;
                 ;++++ TRAZA PUNTO ++++
                 ;
                 NEXT
8C 05 C1                 STY  YLO
AD 14 C7                 LDA  FXLO
8D 03 C1                 STA  XLO
AD 15 C7                 LDA  FXHI
8D 04 C1                 STA  XHI
20 83 C1                 JSR  PLTSUB
                 ;
                 ;++++ INC / DEC COORD Y ++++
                 ;
AD 13 C7                 LDA  UPFLAG
D0 08                    BNE  DECRY
C8                       INP
C0 C8                    CPY  #$C8          ;TOPE MAX. DE Y?
F0 13                    BEQ  NEXLIN
4C 52 C7                 JMP  TESTPT
                 DECRY
88                       DEY
C0 00                    CPY  #$00          ;TOPE MIN. DE Y?
90 0B                    BCC  NEXLIN
                 TESTPT
20 F6 C7                 JSR  POINT         ;PIXEL ILUMINADO
AD 16 C7                 LDA  PTFLAG
D0 03                    BNE  NEXLIN        ;SI, BIFURCAR
4C 2E C7                 JMP  NEXT
                 ;
                 ;++++ EMPIEZA SIGUIENTE LINEA ++++
                 ;
                 NEXLIN
AD 12 C7                 LDA  RTFLAG
D0 1F                    BNE  INCRX         ; COMP. PLAG DER.
AD 14 C7                 LDA  FXLO
38                       SEC
E9 01                    SBC  #$01
8D 14 C7                 STA  FXLO          ;DEC X BYTELO
AD 15 C7                 LDA  FXHI
E9 00                    SBC  #$00
8D 15 C7                 STA  FXHI          ;DEC X BYTEHI

CD 0F C7                 CMP  XMINHI
D0 41                    BNE  MOVE
AD 14 C7                 LDA  FXLO          ;X ALCANZA XMIN?
CD 0E C7                 CMP  XMINLO
D0 39                    BNE  MOVE
60                       RTS                ;FIN RUTINA
                 ;
                 INCRX
AD 14 C                  LDA  FXLO
18                       CLC
69 01                    ADC  #$01          ;INC X BYTELO
8D 14 C7                 STA  FXLO
AD 15 C7                 LDA  FXHI
69 00                    ADC  #$00          ;INC X BYTEHI
8D 15 C7                 STA  FXHI
                 ;
CD 11 C7                 CMP  XMAXHI
D0 22                    BNE  MOVE
AD 14 C7                 LDA  FXLO          ;X ALCANZA XMAX?
CD 10 C7                 CMP  XMAXLO
D0 1A                    BNE  MOVE
                 ;
AD 0B C7                 LDA  XSTLO
8D 14 C7                 STA  FXLO
AD 0C C7                 LDA  XSTHI         ;RESTABLECE PUNTO INICIAL
8D 15 C7                 STA  FXHI
AC 0D C7                 LDY  YST
                 ;
A9 00                    LDA  #$00
8D 13 C7                 STA  UPFLAG
8D 12 C7                 STA  RTFLAG        ;RESTABLECE FLAGS
4C 2E C7                 JMP  NEXT
                 ;
                 ;++++ BUSCA COMIENZO SIG. LINEA ++++
                 ;
                 MOVE
AD 13 C7                 LDA  UPFLAG
D0 IC                    BNE  DOWN          ;COMPRUEBA UPFLAG
88                       DEY                ;Y=Y-1
                 AGAIN1
20 F6 C7                 JSR  POINT
AD 16 C7                 LDA  PTFLAG
D0 04                    BNE  CONT1
C8                       INY
4C BF C7                 JMP  AGAIN1
                 CONT1
EE 13 C7                 INC  UPFLAG        ;PONE A UNO
                 AGAIN2
88                       DEY
20 F6 C7                 JSR  POINT
AD 16 C7                 LDA  PTFLAG
D0 F7                    BNE  AGAIN2
4C 2E C7                 JMP  NEXT
                 ;
                 DOWN
C8                       INY                ;Y=Y+1
                 AGAIN3
20 F6 C7                 JSR  POINT
AD 16 C7                 LDA  PTFLAG
D0 04                    BNE  CONT2
88                       DEY
4C DB C7                 JMP  AGAIN3
                 CONT2
CE 13 C7                 DEC  UPFLAG        ;PONE A CERO
                 AGAIN4
C8                       INY
20 F6 C7                 JSR  POINT
AD 16 C7                 LDA  PTFLAG
D0 F7                    BNE  AGAIN4
4C 2E C7                 JMP  NEXT
                 ;++++ FIN PROGRAMA PRINCIPAL ++++
                 ;
                 ;++++ RUTINA COMPROBACION PUNTO ++++
                 ;
                 POINT
48                       PHA
8A                       TXA
48                       PHA                ;REGS. A PILA
98                       TYA
48                       PHA
                 ;
8C 02 C7                 STY  PXLO
AD 14 C7                 LDA  FXLO
8D 00 C7                 STA  PXLO          ;TRANSF COORDS
AD 15 C7                 LDA  FXHI
8D 01 C7                 STA  PXHI
                 ;
                 ;++++ CALCULA DIRECCION DEL PUNTO ++++
                 ;
AD 01 C7                 LDA  PXHI
8D 04 C7                 STA  PHBHI
AD 00 C7                 LDA  PXLO
29 F8                    AND  #$F8
8D 03 C7                 STA  PHBLO
AD 00 C7                 LDA  PXLO
29 07                    AND  #$07
8D 05 C7                 STA  PREMX
                 ;
AD 02 C7                 LDA  PYLO
29 07                    AND  #$07
8D 07 C7                 STA  PREMY
                 ;
A2 03                    LDX  #$03
                 SHIFT
4E 02 C7                 LSR  PYLO
CA                       DEX
D0 FA                    BNE  SHIFT
AD 02 C7                 LDA  PYLO
8D 06 C7                 STA  PVBYTE
                 ;
A9 00                    LDA  #$00
8D 08 C7                 STA  PROWLO
8D 09 C7                 STA  PROWHI
A2 05                    LDX  #$05
                 FIVE
HB 08 C7                 LDA  PROWLO
18                       CLC
6D 06 C7                 ADC  PVBYTE
8D 08 C7                 STA  PROWLO
CA                       DEX
D0 F3                    BNE  FIVE
A2 06                    LDX  #$06
                 MULT
0E 08 C7                 ASL  PROWLO
2E 09 C7                 ROL  PROWHI
CA                       DEX
D0 F7                    BNE  MULT
                 ;
AD 08 C7                 LDA  PROWLO
18                       CLC
6D 03 C7                 ADC  PHBLO
8D 08 C7                 STA  PROWLO
AD 09 C7                 LDA  PROWHI
6D 04 C7                 ADC  PHBHI
8D 09 C7                 STA  PROWHI
                 ;
AD 08 C7                 LDA  PROWLO
18                       CLC
69 00                    ADC  #MPBLO
8D 08 C7                 STA  PROWLO
AD 09 C7                 LDA  PROWHI
69 20                    ADC  #MPBHI
8D 09 C7                 STA  PROWHI
                 ;
AD 08 C7                 LDA  PROWLO
18                       CLC
6D 07 C7                 ADC  PREMY
85 FB                    STA  PTR
AD 09 C7                 LDA  PROWHI
69 00                    ADC  #$00
85 FC                    STA  PTR+1
                 ;
A9 01                    LDA  #$01
8D 0R C7                 STA  PBPOS
38                       SEC
A9 07                    LDA  #$07
ED 05 C7                 SBC  PREMX
F0 07                    BEQ  BITON
AA                       TAX
                 POWER
0E 0A C7                 ASL  PBPOS
CA                       DEX
D0 FA                    BNE  POWER
                 ;
                 ;++++ COMPRUEBA BIT ON ++++
                 ;
                 BITON
A0 00                    LDY  #$00
B1 FB                    LDA  (PTR),Y       ;CARGA CONTENIDO DIRECCION
2D 0A C7                 AND  PBPOS
8D 16 C7                 STA  PTFLAG        ;ALMACENA RESULTADO
                 ;
68                       PLA
A8                       TAY
68                       PLA                ;RESTAB. REG. DE PILA
AA                       TAX
68                       PLA
                 ;
60                       RTS
```

Calidad garantizada

Softsel es el más importante distribuidor de hardware y software en todo el mundo y cumple una valiosa misión: el control de calidad de los productos

Softsel proporciona a los minoristas de software para ordenador un servicio sumamente valioso. En una época en la que los paquetes de software, algunos de ellos muy caros, están saliendo a la venta cada vez más rápido, un comerciante se encuentra con el problema de evaluar personalmente cada producto nuevo, lo que representa un consumo de tiempo y de dinero, o, de lo contrario, confiar en una evaluación apresurada, lo que podría significar "clavarse" con algunos paquetes invendibles y costosos. El servicio que proporciona Softsel consiste en liberar al minorista de este elemento de riesgo, haciendo una profunda evaluación de producto a todos los paquetes que la empresa incorpora a su lista de software.

Simon Rhodes, director de marketing para Gran Bretaña, explica el procedimiento: "El paquete es examinado primero por nuestro equipo técnico para comprobar su sencillez de uso para el usuario: si está bien programado, bien documentado, si posee buenos gráficos, etc. Luego pasa a nuestro departamento de marketing y ventas, donde se decide si tendrá una buena promoción". Según las estimaciones de Softsel, en un semestre reciente sólo el 10 % de cerca de 700 paquetes se aceptaron para su inclusión en el catálogo de la empresa.



Herb Blumstein, director gerente

Simon Rhodes, director de marketing

Ian McKinnell

Este control de calidad, junto con los incentivos adicionales de acuerdos de ventas y devoluciones y la necesidad de tratar con un solo proveedor, hacen que una empresa como Softsel sea una propuesta atractiva para un minorista de software. Indudablemente, un minorista podría comprar los paquetes a un precio más reducido directamente de un fabricante, pero la capacidad de Softsel para ofrecer descuentos en función de su potencial de compras en grandes volúmenes hará que la diferencia sólo sea marginal. Simon Rhodes señala: "Aunque un comerciante puede obtener un precio mejor, comprarle directamente a los fabricantes a la larga le costará más, porque tendrá que tratar con cientos de personas diferentes en vez de con una sola empresa".

Orígenes norteamericanos

Softsel fue fundada en 1980 por Robert Leff y David Wagman, quienes habían sido compañeros de trabajo en el departamento de proceso de datos de Transaction Technology, subsidiaria de la gigantesca empresa financiera Citicorp. La creencia de Leff y Wagman de la necesidad de una empresa de las características de Softsel se demostró mediante su rápida expansión. A los cuatro años de su creación, Softsel tiene 350 empleados en todo el mundo y el movimiento total internacional de la empresa durante el último año comercial alcanzó la impresionante cifra de 87 millones de dólares. Sólo en Estados Unidos la empresa posee cuatro enormes almacenes (en Atlanta, Chicago, Los Ángeles y Nueva York), que proveen una gama de 4 500 paquetes a establecimientos minoristas de todo el país.

La penetración de la empresa en el mercado británico comenzó en septiembre de 1982. En abril del año siguiente se creó una subsidiaria, Softsel Computer Products. La filial de Gran Bretaña tiene su base en Feltham, cerca del aeropuerto de Heathrow, y suministra más de 2 500 productos diferentes a minoristas de toda Europa y Oriente Medio.

El futuro de la empresa parece brillante. La subsidiaria británica tiene planeado incrementar el porcentaje de su catálogo de software destinado a paquetes de gestión, que ya representa más de la mitad del catálogo. No obstante, la empresa no tiene intenciones de restarle importancia a la otra principal área de la producción de software, la de los programas de juegos.

Asimismo, Softsel pretende aumentar su participación en el mercado europeo. Ya se ha creado una subsidiaria en Alemania, con oficinas centrales en Munich, y está previsto el establecimiento de subsidiarias en Francia e Italia.

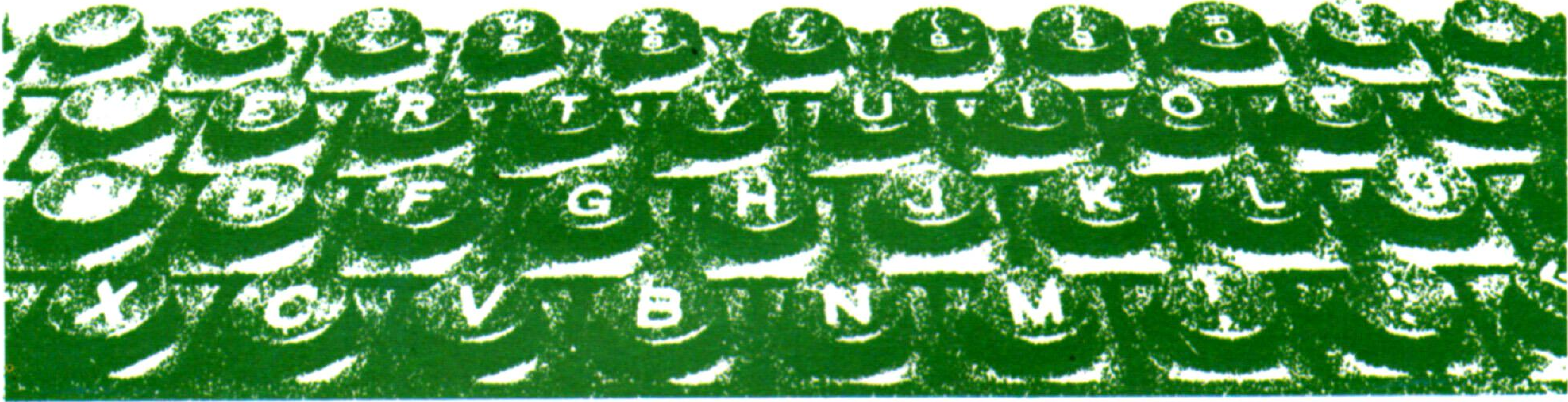
mi COMPUTER

Con el fascículo anterior se han puesto a la venta las tapas correspondientes al cuarto volumen.

El juego de tapas va acompañado de un sobre con los transferibles, numerados del 1 al 8, correspondientes a los volúmenes de que consta la obra; esto le permitirá marcar el lomo de cada uno de los volúmenes, a medida que aumente su colección.

Para encuadernar los 12 fascículos que componen un volumen, es preciso arrancar previamente las cubiertas de los mismos.

No olvide que, antes de colocar los fascículos en las tapas intercambiables, debe usted estampar el número en el lomo de las mismas, siguiendo las instrucciones que se dan a continuación:



1 Desprenda la hojita de protección y aplique el transferible en el lomo de la cubierta, haciendo coincidir los ángulos de referencia con los del recuadro del lomo.

2 Con un bolígrafo o un objeto de punta roma, repase varias veces el número, presionando como si quisiera borrarlo por completo.

3 Retire con cuidado y comprobará que el número ya está impreso en la cubierta. Cúbralo con la hojita de protección y repita la operación anterior con un objeto liso y redondeado, a fin de asegurar una perfecta y total adherencia.

Editorial Delta

Cada sobre de transferibles contiene una serie completa de números, del 1 al 8, para fijar a los lomos de los volúmenes. Ya que en cada volumen sólo aplicará el número correspondiente, puede utilizar los restantes para hacer una prueba preliminar.

10049

9 788485 822836